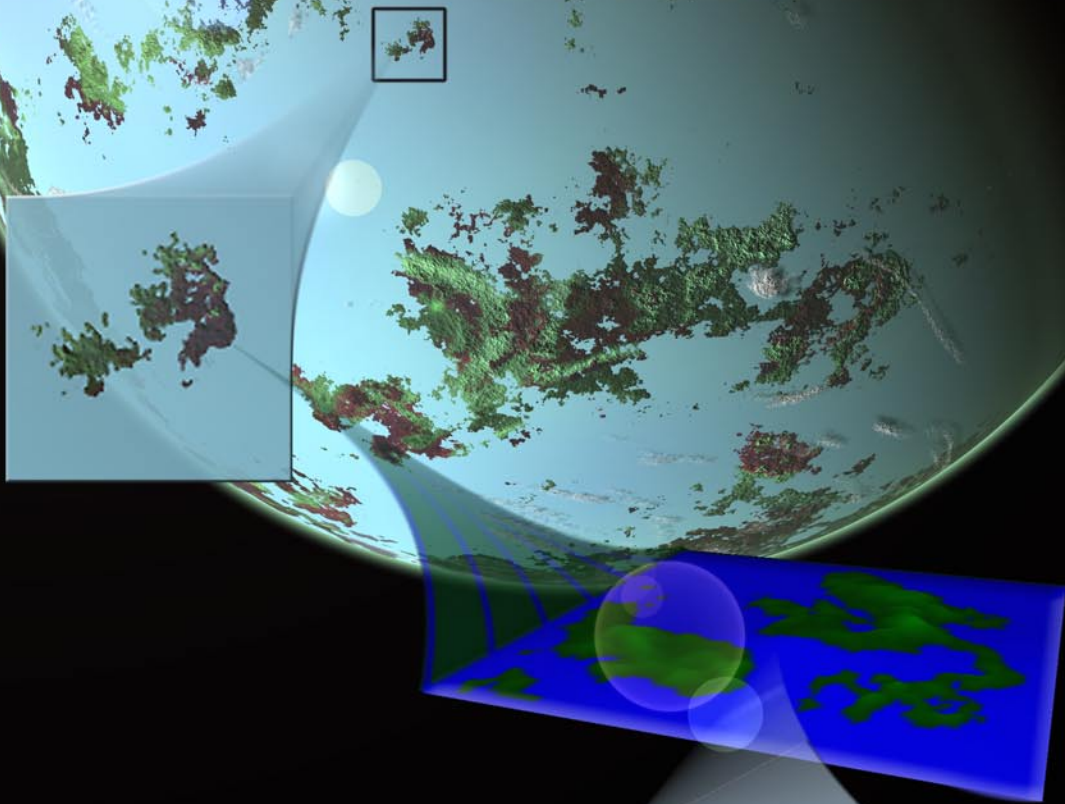


Sniping-Jacks Wrap-Tutorial

Version 0.9



Vorwort

In diesem Tutorial werdet ihr erfahren, wie man von Grundauf eine eigene **OFF-Insel** erstellen kann. Alle dafür notwendigen Tools werden so genau wie möglich beschrieben, und Schritt für Schritt anhand von Beispiel-Inseln erklärt. Dieses Tutorial ist auf die Verwendung des neuen **WrpTool** von Nekromantix abgestimmt. Andere Tools finden hier keine Erwähnung. Natürlich wird auch auf das Erstellen der Insel-Rohdaten per **Terragen und Wilbur** eingegangen. Ihr werdet lernen, wie man schon bei der Rohdatenerstellung seine Vorstellungen ganz gezielt umsetzen kann, und das zukünftige Terrain schon jetzt für **Flughäfen, Brücken, Dörfer** und **Straßen** vorbereitet.

In diesem Tutorial wird nicht auf alle Möglichkeiten in WrpTool eingegangen, das würde den Rahmen sprengen. Für die fast zwangsläufig auftauchenden Fragen werden sich in den einschlägigen Foren bei **AEF**, und sicherlich bald auch auf **MAPFACT** kompetente Ansprechpartner finden.

Um die **Navigation** im Tutorial zu erleichtern, habe ich **Links** eingebaut. Das Inhaltsverzeichnis ist interaktiv, und darüberhinaus stellt es euch den Zoomfaktor gleich auf lesefreundliche Darstellung ein. Wer auf eine **Überschrift** (die megafette Schrift) klickt, kommt wieder zum **Inhalt** zurück. **Rote, fette** Textstellen bezeichnen immer eine **Funktion** im jeweiligen Programm, **schwarze und fette** Textstellen sind reine Hervorhebungen und dienen nur dem schnellen **wiederfinden/erkennen**. **Hellblaue, magere** Textstellen sind Hinweise, die eine Quelle im Internet symbolisieren. Einfach das Schlagwort in die Suchmaschine eingeben, und ggf. zum Ziel weiterhangeln.

Die **Bilder** im Tut sind manchmal stark verkleinert eingesetzt worden - zoomt einfach etwas stärker hinein, dann sind die Details wieder gut erkennbar. Ich habe die Bilder mit 150dpi eingebunden, ein Druck sollte möglich sein.

Bitte werft immer auch einen Blick ins ausgezeichnete **englische Manual**, dort erhaltet ihr die Infos, auf die ich in diesem Tutorial nicht eingehen konnte. Außerdem könnte mir auch mal ein Fehler unterlaufen sein, oder einfach etwas ungeschickt erklärt haben. Dort „werdet ihr dann weitergeholfen“ ;-)

Inhalt

<i>Einleitung</i>	5
<i>DL-Adressen aller Tools</i>	5
<i>Grundsätzliche Überlegungen</i>	6
<i>Verschiedenes und Nützliches</i>	8
<i>Start mit Terragen</i>	10
<i>Von Terragen zu Wilbur</i>	11
<i>Start in Wilbur</i>	12
<i>Installation von WrpTool</i>	17
<i>Vorbereitungen u. erste Schritte</i>	20
<i>3D-View und Mini-Map</i>	25
<i>Der Elevation-Mode (F6)</i>	26
<i>Das Elevation-Tool (Strg+“E“)</i>	27
<i>Der Texture-Mode (F7)</i>	32
<i>Der Textur-Browser (Strg+“T“)</i>	32
<i>Der Texture-Finder</i>	33
<i>Create Transitions</i>	34
<i>Der Object-Mode (F5)</i>	39
<i>Der Object-Browser (Strg+“O“)</i>	39
<i>Der Object-Finder</i>	40
<i>Der ODOL-Explorer</i>	40
<i>Der Object-Inspector (Strg+“I“)</i>	43
<i>Der Object-Placer (Strg+“P“)</i>	44
<i>Das Road-Tool</i>	46

<i><u>Das Objekt-Tool (Strg+“R“)</u></i>	<i><u>49</u></i>
<i><u>Wälder mit dem Region-Tool</u></i>	<i><u>51</u></i>
<i><u>Der Cell Selector</u></i>	<i><u>52</u></i>
<i><u>KI-tauglicher Flugplatz</u></i>	<i><u>54</u></i>
<i><u>Seehafen bei Ebbe und Flut</u></i>	<i><u>56</u></i>
<i><u>Kartennamen</u></i>	<i><u>57</u></i>
<i><u>Objekte und Texturen ersetzen</u></i>	<i><u>58</u></i>
<i><u>Im-/Export von Inselteilen</u></i>	<i><u>59</u></i>
<i><u>Real-Maps für WT</u></i>	<i><u>60</u></i>
<i><u>WrpTool „frisieren“</u></i>	<i><u>62</u></i>
<i><u>Insel für OFP fertigstellen</u></i>	<i><u>65</u></i>
<i><u>Kleine „Seltsamkeiten“</u></i>	<i><u>69</u></i>
<i><u>Schlussbemerkungen</u></i>	<i><u>70</u></i>



Einleitung

Hiermit begrüße ich nun alle recht herzlich, die sich vorgenommen haben, die Welt zu verbessern – natürlich erstmal nur in OFP! ;-). Mancher mag sich beim ersten Blick auf soviel Text erschlagen fühlen. Aber ich verspreche, dass es nur halb so schlimm sein wird, wie es im ersten Moment aussieht. Ich gehe lediglich etwas ausführlicher an die Thematik heran, und bemühe mich dabei, möglichst viel Fragen schon „im Keim zu ersticken“. Und wer sich an meine Anleitung Schritt für Schritt hält, ist fast im Handumdrehen im Besitz eines eigenen Inselreiches. Solltet ihr dennoch mal „hängen“, lasst euch nicht entmutigen. In den Foren von [AEF](#), und vielleicht demnächst auch [MAPFACT](#) (um nur zwei der Wichtigsten zu nennen) findet ihr sicher eine Antwort auf eure Fragen.

Ich wünsche mir jedenfalls, dass der Neuling dieses Tutorial ebenso nützlich finden wird, wie der Fortgeschrittene, der darin den einen oder anderen Hinweis entdecken könnte. Und nun wünsche ich euch viel Spaß beim Wrappen! :-)

Links

OK, schreiten wir zur ersten Tat, werfen unsere Compis an, und saugen alle benötigten Programme aus dem Netz:

Terragen:	http://www.planetside.co.uk/terragen...
Wilbur:	http://www.ridgenet.net/...
WrpTool:	http://flashpoint.nekromantix.com/...
PB0 Decryptor v1,5:	http://www.operationflashpoint.d2.cz/...
PB0-Packer:	http://www.operationflashpoint.d2.cz/...
Texview:	http://www.operationflashpoint.d2.cz/...
Binarize:	http://www.codemasters.com/...

Ich habe nur die Startaddy angegeben, da sich die genauen Ziele allzuoft ändern, aber ich bin sicher, ihr findet alles nötige auch so.

Außerdem wird Photoshop, oder ein vergleichbares Bildbearbeitungsprogramm benötigt, um für eigene Previews die Screenshots ins richtige xxx.BMP-Format wandeln zu können. Wer sich für diese Arbeit nicht interessiert, braucht auch kein Photoshop.

Grundsätzliches

Bevor wir beginnen, sollten wir uns ein paar grundsätzlichen Überlegungen zuwenden. Es ist wichtig, dass wir eine klare Vorstellung von unserer zukünftigen Insel entwickeln. Je genauer das Bild, desto gezielter und erfolgreicher die Umsetzung. Dazu müssen wir natürlich wissen, was WT (WrpTool) kann, bzw. was es nicht kann. Zuerst, was WT **nicht** kann:

Nicht-Quadratische Inselgrößen

Insel-Höhen >1474m

Insel-Größen >1024² Zellen (z. Vergl.: Novoga =256²)

Auf Win9x-Systemen laufen, W2k oder XP wird benötigt!

Zaubern und Kaffeekochen

Ansonsten haben wir mit keinen bedeutenden Einschränkungen zu kämpfen. Wir könnten ohne weiteres eine Insel erstellen, auf der nur vier Texturkacheln Platz fänden. Wichtig für WT, und vor allem für OFP ist dabei nur, dass die Insel-Größe ein vielfaches von 2² ist. Das sieht dann so aus: **2²; 4²; 8²; 16²; 32²; 64²; 128²; 256²; 512²**, und schließlich **1024² Zellen**; die Mutter aller Inseln – über 50 km Länge und Breite!!! Es wären rein Technisch zwar auch 2048² Zellen möglich, doch OFP würde dabei einfach „Tschüsssss“ sagen und sich kommentarlos verabschieden. Behalten wir also **1024² Zellen** als unsere **Obergrenze**, und **32² Zellen** als sinnvolle **Untergrenze** im Gedächtnis. 32² Zellen entsprechen einer Kantenlänge von ~1,6km. Wer mit so wenig Platz dennoch etwas damit Anfangen kann, möglich ist es. In meinem Tutorial wird jedoch eine Untergrenze von 64² Zellen empfohlen, weil wir ja die Rohdaten für die Insel in Wilbur erstellen wollen., und dort machen noch kleinere Inseln wenig Freude beim Zeichnen.

So weit, so gut. Ihr wisst nun darüber Bescheid, was Ihr NICHT dürft. Doch ich denke, dass die Einschränkungen unseren Tatendrang nicht wirklich einschränken werden – oder? Eben, das denke ich doch auch. Überlegen wir uns also nun, wie groß unsere Insel werden soll, ob es nur eine, oder mehrere sein sollen, wie die Infrastruktur beschaffen sein soll, welche klimatischen Bedingungen dort vorherrschen sollen, und schlussendlich, ob es einen **Flughafen** geben soll. Diese Frage ist insofern von weitreichender Bedeutung, da für das Scripten von eigenständigen **Starts** und **Landungen** eine **ILS-Datenstruktur** für die **KI** vonnöten ist.

Wer einen KI-tauglichen Flughafen braucht, sollte unbedingt auch einen Blick ins englische Manual werfen, da ich bisher noch keine Gelegenheit hatte, die etwas komplexe Thematik in der Praxis auszuprobieren.

Folgendes wird erst dann wichtig, wenn ihr das Tut schon einmal durch habt, und ihr schon konkretere Pläne schmiedet – beim ersten mal ist das noch nicht so wichtig. Sobald Ihr also eure Ideen gesammelt habt, legt ihr eine **Readme** darüber an, und haltet darin eure Vorstellung fest. Welchen Zweck soll die Map erfüllen, und wie wollt ihr das erreichen. Ein paar Beispiele: soll die Map einen Original-Schauplatz wiedergeben, oder hat die Fantasie freien Lauf? Soll das trotzdem sehr natürlich aussehen, oder darf die Insel auch ein wenig „abgefahren“ sein? Soll die Map noch ausreichend Rechenreserven bieten, oder wollt ihr euch Landschaftlich mit allem austoben, was die Tools hergeben? Das ganze ist insofern von Bedeutung, weil es darüber entscheidet, mit welchem Tool ihr beginnen werdet. Fürs erste aber solltet Ihr mit beiden Tools (Terragen & Wilbur) einmal gearbeitet haben, damit ihr überhaupt in der Lage seid, später die richtige Entscheidung zu treffen. Wenn Ihr dann eure Erfahrungen gesammelt habt, werdet ihr im Schlaf wissen, was ihr benötigt.

Ein weiterer Punkt betrifft die **Texturierung**. Hier gibt es Einschränkungen, die nicht von Wrptool verursacht werden, sondern in der Verfügbarkeit der Texturen begründet sind. Um diesen Mangel wenigstens zu lindern, habe ich einen eigenen Textursatz entwickelt. Ihr findet ihn ebenfalls zum DL auf Mapfact bereitliegen. Er nennt sich „**Jack_256.ace**“, und ich empfehle euch, das Teil ebenfalls zu saugen. Dieses Set hat zwar auch Einschränkungen, aber im Gegensatz zu den Original-Texturen sind diese ganz klar umrissen und definiert. Es sollte euch also gerade am Anfang leichter fallen, mit diesem Set zu arbeiten, und erste Erfolgserlebnisse einzuheimsen. Ohne gleich wieder nicht einlösbare Versprechen abgeben zu wollen, beabsichtige ich in der Zukunft, dieses Set Stück für Stück zu erweitern, und somit mehr und mehr Einschränkungen beim Texturieren zu eliminieren. Da „Jack“-Set ist von den Resistance-Texturen abgeleitet, und daher bedingt kompatibel. Mischen sollte man diese beiden dennoch nicht so ohne weiteres, da ich die Auflösung reduziert habe. Doch ist diese immer noch doppelt so groß wie bei den CWC-Texturen (CWC= Cold War Crisis = Everon, Malden, Dessert Island und Kolgujev). Zusätzlich habe ich noch ein paar kleine Bonus-Texturen hinzugefügt – z.B. zwei kleine Buschpisten für versteckte Operationen ;-)

Verschiedenes...

...und **Nützliches**, bevor wir beginnen. Diese Seite ist sozusagen der Sammelbehälter für all die Dinge, die einem erst im Laufe der Arbeit einfallen, aber für den Leser schon von Beginn an wichtig, oder wenigstens nützlich sein könnten. Zum Beispiel dies: Nützlich ist (sofern möglich) häufiges speichern. Ist nix neues, schon klar. Trotzdem passiert's halt immer wieder, dass man gerade im Eifer des Gefechts auf die rüde Tour daran erinnert wird, was man längst schon hätte tun sollen ;-)

Es ist auch hilfreich, sich vor Beginn klarzumachen, welche **Version** von OFP nötig sein wird. Wer z.B. auf alles verzichten kann, was OPF-Resistance (dazu zählen natürlich auch viele Addons!) benötigen würde, dessen Insel ist bestens für anspruchsvolle Scripts und/oder ationlastige Maps geeignet. Es bleibt für den Scripter einfach viel mehr an Rechenleistung dafür übrig. Wer beim Wrappen auf so etwas achtet, sollte das dann auch in der Readme ausweisen - die Mapper werden's ihm danken!

Achtet bitte auch darauf, dass ihr eure Insel-Tests in OFP möglichst mit dem **Gelände-Detailgrad** „Normal“ überprüft. Das dürfte der Einstellung von ca. 95% aller User entsprechen. Weicht man davon ab, könnte es zu Problemen kommen, sobald der User nicht dieselbe Einstellung benutzt.

Oben genannte Einstellungen, sowie eine **Charakter-Beschreibung** der Insel sollten in einer Readme aufgeführt sein, die der User VOR dem DL einsehen kann. Ein Screen ist da übrigens nicht sehr hilfreich - eine Beschreibung der wichtigsten Insel-Parameter nützt dem User weitaus mehr. Teilt hier allen mit, was einen erwartet, wenn man sich die Insel saugen möchte, z.B.: Benötigte Version, Detailgrad, Anzahl der Inseln, Vegetation, bergig oder flach, Flüsse oder nicht, alte Gebäude oder modern, Flughafen oder nicht, usw.usw. Dann kann jeder schon vorher einschätzen, ob die Insel dem entspricht, was man grade sucht. Denn wer saugt schon ein paar MB, und stellt erst hernach enttäuscht fest, dass das Teil nicht im geringsten seinen Anforderungen gerecht wird?!

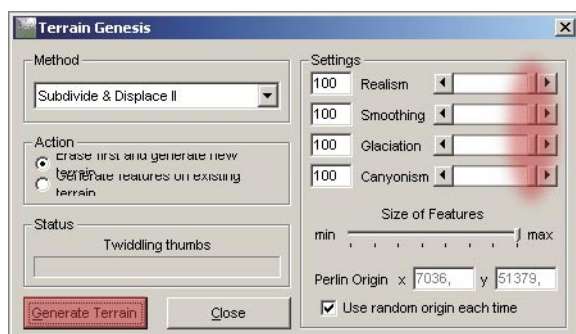
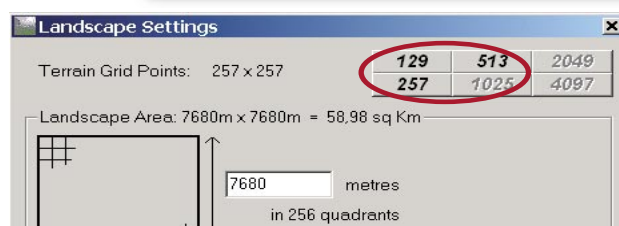
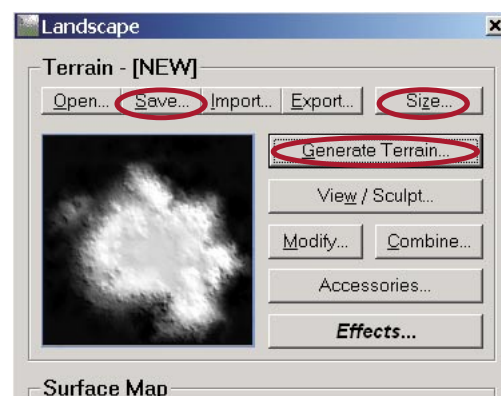
Und ganz wichtig, wenn ihr nach dem ersten „Durchlauf“ euer erstes ernsthaftes Projekt beginnt: Mit Terragen/Wilbur lieber einmal zuviel **experimentieren und üben**, als einmal zu wenig. Hat man erstmal seine Insel texturiert, und mit Objekten ausgestattet, ist ein Fehler, der zu einem kompletten Neustart zwingt, sehr, sehr frustrierend! Folgt dem Tutorial erstmal bis zum Ende, und führt die dort gegebenen Anweisungen nur so weit aus, bis ihr die Materie im Griff habt. Erst wenn diese erste Test-Insel fertig ist, und auch in OFP tadellos funktioniert, dann macht ihr euch über euer eigentliches Vorhaben her. Gerade die Rohdatenerstellung in Terragen und Wilbur ist das „A und O“ für eine gute Insel, was deren Geländebeschaffenheit betrifft. Wer z.B. Wilbur voll auslotet (ein ausführliches Tut liegt neben Wilbur zum DL bereit), kann mehr als 95% seiner Vorstellung schon dort umsetzen. Aber um ebendiese Vorstellung erstmal voll entwickeln zu können, solltet ihr euch der Möglichkeiten in WrpTool erstmal bewusst werden. Daher bringt es nichts, schon von Anfang an perfekte Ergebnisse erzielen zu wollen. Herumexperimentieren ja – aber so, dass Fehler dabei noch keine Rolle spielen. Und Fehler sind das Beste, was euch dabei passieren kann, denn dabei lernt man am meisten!

Manche Screenshots stammen schon aus der neuen Version von WrpTool 0.950. Lasst euch davon nicht irritieren, denn die Unterschiede sind nur gradueller Natur. Aus Zeitgründen wurde dieses Tutorial ein „Misch-Masch“ aus den Versionen 0.906 und 0.950. Manche Neuerungen habe ich jedoch schon jetzt beschrieben, oder wenigstens angeschnitten. Wer dieses Tut durcharbeitet, kann getrost schon das neue WrpTool installieren, die Prozedur ist die gleiche. Auch sonst sind die Unterschiede nie so groß, dass man ins Schleudern geraten würde. Zudem werde ich nach Veröffentlichung einen Thread auf MAPFACT einrichten, in dem ihr dann eure wichtigsten Fragen schonmal posten könnt. Zudem empfehle ich dringend jedem Wrapper, auch die WrpTool-Threads im AEF-Forum zu lesen. Dort sind u.a. die Entwickler von WrpTool zugegen, und leisten vorzüglichen Support zu ihrem Tool. Doch BITTE! seid so anständig, zuerst die Suchfunktion zu benutzen, bevor ihr schon gestellte Fragen nochmal neu stellt. Das sollte zwar selbstverständlich sein, dennoch passiert es leider immer wieder.

Wenn ihr mich wegen „sachdienlicher Hinweise“ zum Tutorial kontaktieren wollt, dann mailt einfach an: **sniping-jack@cool.ms**

Start in Terragen

Terragen eignet sich vor allem für diejenigen, die mit Inselgrößen von **128²**, **256²** und **512²** arbeiten möchten. Der Vorteil, mit Terragen zu arbeiten ist, dass Terragen ausgesprochen **natürliche Terrains** erzeugen kann – und dies in kürzester Zeit. Diejenigen, die definitiv noch größere, oder kleinere Inseln erstellen möchten, können gleich zu „**Wilbur**“ jumpen. Ich empfehle dennoch dringend, die Roh-Insel in Terragen zu beginnen, da Wilbur nicht speichern kann. Wer Terragen noch nicht installiert haben sollte, möge dies jetzt tun. Es ist absolut unproblematisch, es gibt nichts Besonderes zu beachten. Zuerst stellen wir die Größe bei „**Size**“ (erst ab Vers. 0.9x möglich) ein. Danach klicken wir auf das „**Wasser**“-Symbol links in der Leiste, und ändern im neuen Fenster die Meereshöhe von **-300** auf **0** Meter. Danach ist für uns nur das Fenster „**Landscape**“ mit der Schwarz/Weiß-Grafik wichtig – dort generieren wir unsere Insel, indem wir auf „**Generate Terrain**“ klicken. Um zu guten Ergebnissen zu gelangen, stellen wir die **Regler** im Fenster „**Terrain Genesis**“, wie in der Abbildung gezeigt, ein. Nun klicken wir so oft auf „**Generate Terrain**“ im „**Terrain Genesis**“-Fenster, bis wir einen möglichst hellen Bereich im Vorschaufenster (bei „**Landscape**“), sowie möglichst dunkle Bereiche an den Fensterrändern sehen. Die Ab-

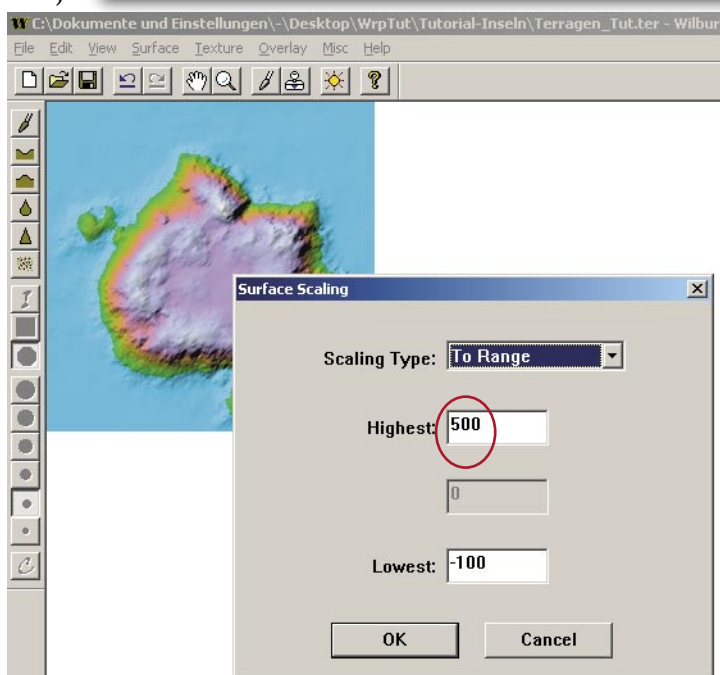
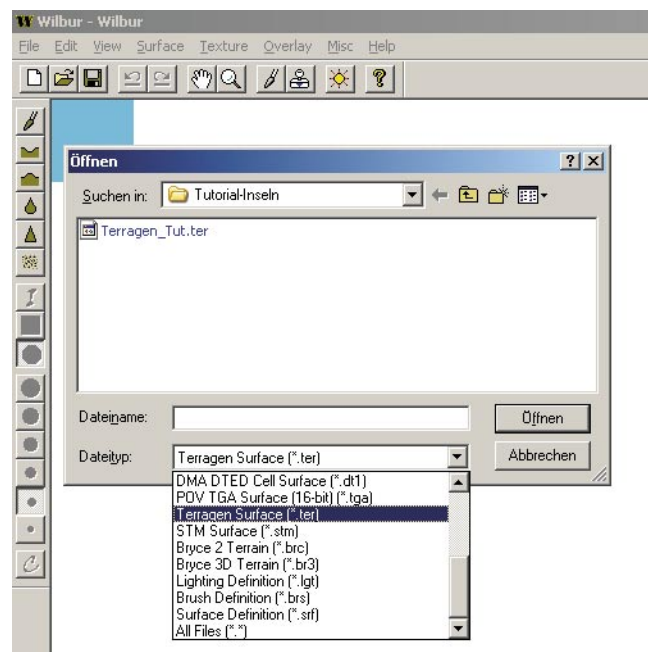
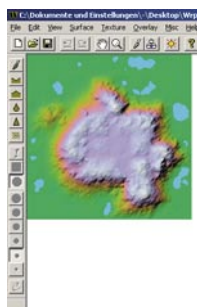


bildung oben zeigt, wie in etwa das Ganze aussehen könnte/sollte. Klappt das nicht, dann den „Size of Features“-Regler etwas zurücknehmen. Zur Erklärung sei gesagt, dass helle Bereiche als hoch, und dunkle Bereiche als niedrig interpretiert werden. Nehmen wir an, im Fenster ist in der Mitte nun ein möglichst heller Bereich, und am Rand alles schön Schwarz, dann können wir uns schon ans Speichern („**Save**“ oben rechts) machen, nachdem wir (**wichtig!**) das Fenster „Terrain Genesis“ geschlossen haben. Geben wir unserem Rohdatensatz einen einprägsamen Namen, z.B. „**My-World_256.ter**“, und speichern das File einen eigenen Ordner, den wir für unsere Rohdaten anlegen sollten. Die „...256...“ im Namen bezieht sich auf unsere zukünftige Inselgröße.

Terragen>Wilbur

Die Installation von Wilbur gestaltet sich denkbar einfach: Es gibt nämlich keine. Entpacken und starten – das wars schon. Jedoch sind zwei Dinge bei Wilbur **wichtig**: Erstens kann man seine eigene, oder von Terragen geöffnete, und dann veränderte Map NICHT sichern! Abspeichern schon, aber sobald man diese wieder lädt, kommt im Grunde nur Mist dabei raus. Und zweitens stürzt Wilbur zu allem Überfluss auch ganz gerne manchmal ab. :-(Etwas besser sind nur jene dran, die ihre Insel in Terragen erstellt haben, denn die lässt sich wieder neu laden. Allerdings sind ggf. schon vorgenommene Änderungen futsch.

An dieser Stelle klinken sich all jene ein, die soeben mit Terragen eine Rohdatei erzeugt hatten. Wer nicht, springt zu „**Start in Wilbur**“ weiter. Alle Terragen-User öffnen (**File > Open**) jetzt in Wilbur die soeben erzeugte „**MyWorld_256.ter**“ (Bei mir heißt diese eben „Terragen_Tut.ter“). Dazu muss im Öffnen-Dialog „**Terragen Surface (.ter)**“ eingestellt sein. Das Ganze sollte dann ungefähr wie rechts gezeigt aussehen. Im kleinen Bild links seht Ihr, wie die geöffnete Datei in Wilbur erscheint. Wie man jedoch deutlich erkennen kann, ist statt eines Meeres nur ein „Pfützen-Teppich“ vorhanden. Dem werden wir jetzt wie folgt abhelfen: Handelt euch im Dialogfeld „**Surface**“ über „**Point Process**“ zu „**Offset**“ durch. In dem Dialogfenster gebt ihr dann einen Wert zwischen -10 und -20 ein. Ihr könnt auch höhere Werte eingeben, entsprechend steigt dann der



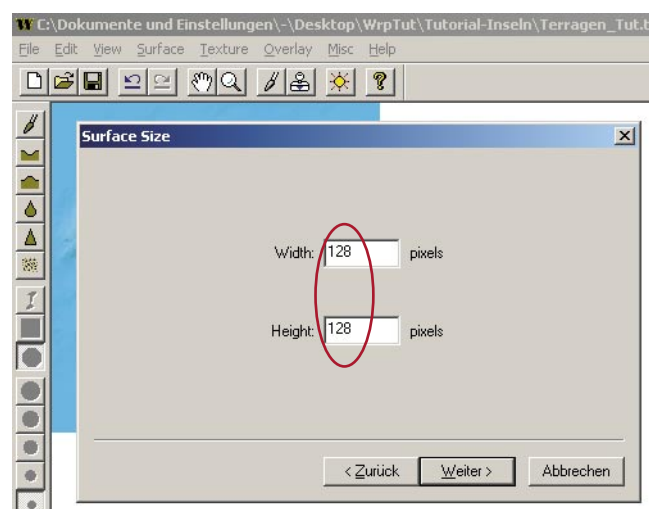
Meeresspiegel. Nun sieht unsere Map schon etwas besser aus, ist aber noch nicht ganz brauchbar. Denn wenn ihr mal die Maus über die Map bewegt, und dabei einen Blick auf das Koordinatenfenster ganz rechts unten in Wilbur werft (3. Wert nach Doppelpunkt = Höhe/Tiefe in Metern), dann fällt auf, dass unsere Insel noch erbärmlich flach ist. Aber auch dem kann abgeholfen werden. Klickt euch nochmals über „**Surface** > **Point Process**“ zu „**Scale**“ durch. Hier (siehe 2. Bild vorherige Seite) können wir nun die Höhengrenzen unserer Insel bestimmen. Meine Beispielinsel wäre hier also 500m hoch, allerdings ist das Meer nicht 100m tief. Warum WrpTool trotzdem nur auf 2,3m Wassertiefe kommt, das weiß der Geier. So, im Grunde war's das schon, unsere Insel ist soweit fertig. Wer jetzt schon möchte, kann zu „**Speichern**“ springen, und seine Insel anschließend in WrpTool laden. Wer aber seine Insel noch ein wenig modifizieren, bzw. von Grund auf neu gestalten möchte, der liest hier weiter.

Start in Wilbur

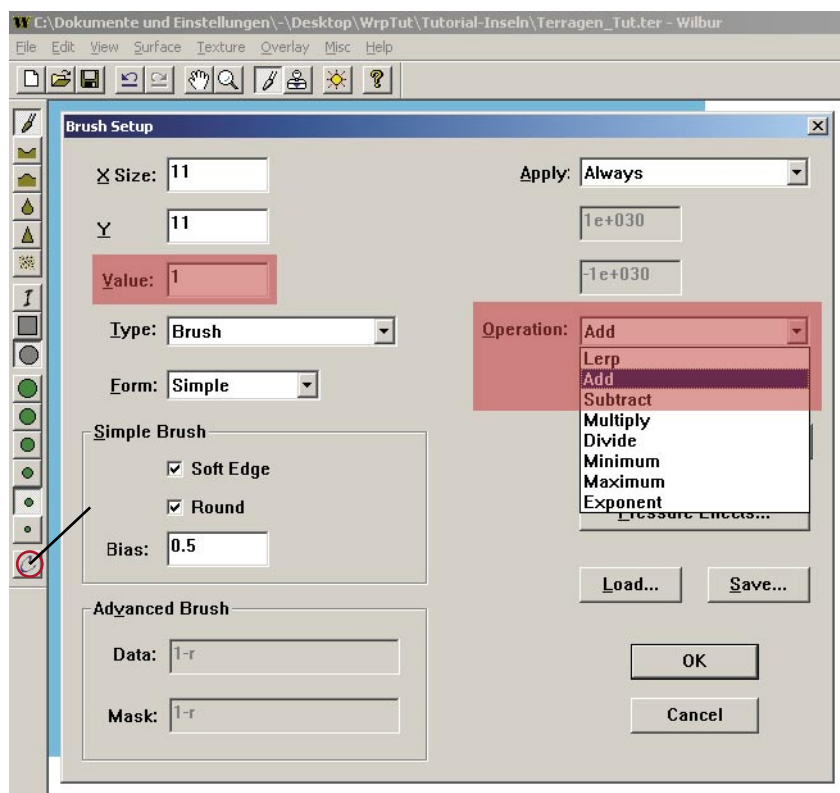
Wer den Teil mit Terragen übersprungen haben sollte, öffnet jetzt ebenfalls Wilbur. Nach dem ersten Öffnen sollte eine blaue Fläche links oben zu sehen sein. Aber um sicher zu gehen, dass jeder auch wirklich die Inselgröße erhält, die er sich gerade wünscht, klicken wir auf „**File**“ > „**New**“ und geben eine Inselgröße ein. In diesem Beispiel **128** Zellen Breite und Höhe.

Nochmal: Hier immer zwei gleiche Zahlen aus der oben genannten **Zahlenreihe** eingeben! Das danach erscheinende Fenster bestätigen wir einfach mit „**OK**“. Prima, nun haben wir eine blaue Fläche, auf der wir uns austoben können. Bevor wir jedoch loslegen, müssen wir den Meeresboden von 0 Meter auf etwa -10 Meter senken. Dazu öffnen wir das Menü „**Surface**“ > „**Point Process**“ > „**Offset**“ und

geben dort im Dialogfenster „-10“ ein. Mit „**OK**“ bestätigen und fertig. Machen wir uns als nächstes mit den wichtigsten Tools vertraut. Das Lupensymbol oben dürfte klar sein, und zum Verkleinern einfach gleichzeitig „**Shift**“ drücken. Vergrößert also eure Map ein wenig – sagen wir um zwei „Clicks“ mit der Lupe.... Gut. Aktivieren wir als nächstes das „**Brushtool**“ in der

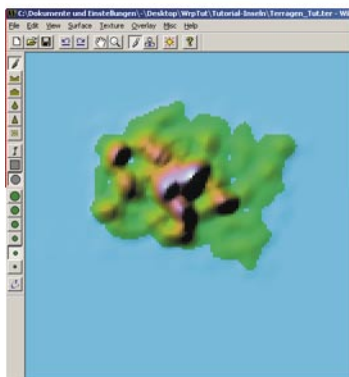
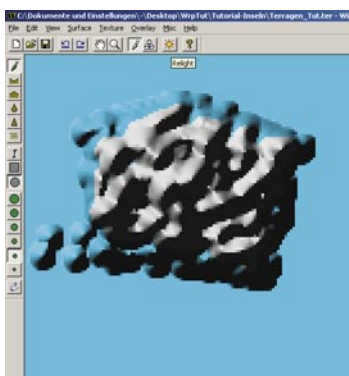


linken Leiste ganz oben. Unmittelbar danach klicken wir auf das letzte Symbol links ganz Unten. Darauf öffnet sich das „**Brush-Setup**“ für den Pinsel (siehe rechts). Für uns sind nur die rot unterlegten Funktionen von Belang. Wer mit weiteren Funktionen experimentieren möchte, dem steht nichts im Wege. Aber darauf hier und jetzt einzu-



gehen, würde den Rahmen diese Tuts sprengen. Besprechen wir also die drei wichtigen Eigenschaften, die unserem Pinsel damit zugewiesen werden können. „**Add**“ und „**Substrakt**“ sollten selbsterklärend sein. Bei jedem Klick wird das Terrain um den Betrag gehoben oder gesenkt, der im Feld „**Value**“ eingestellt ist.

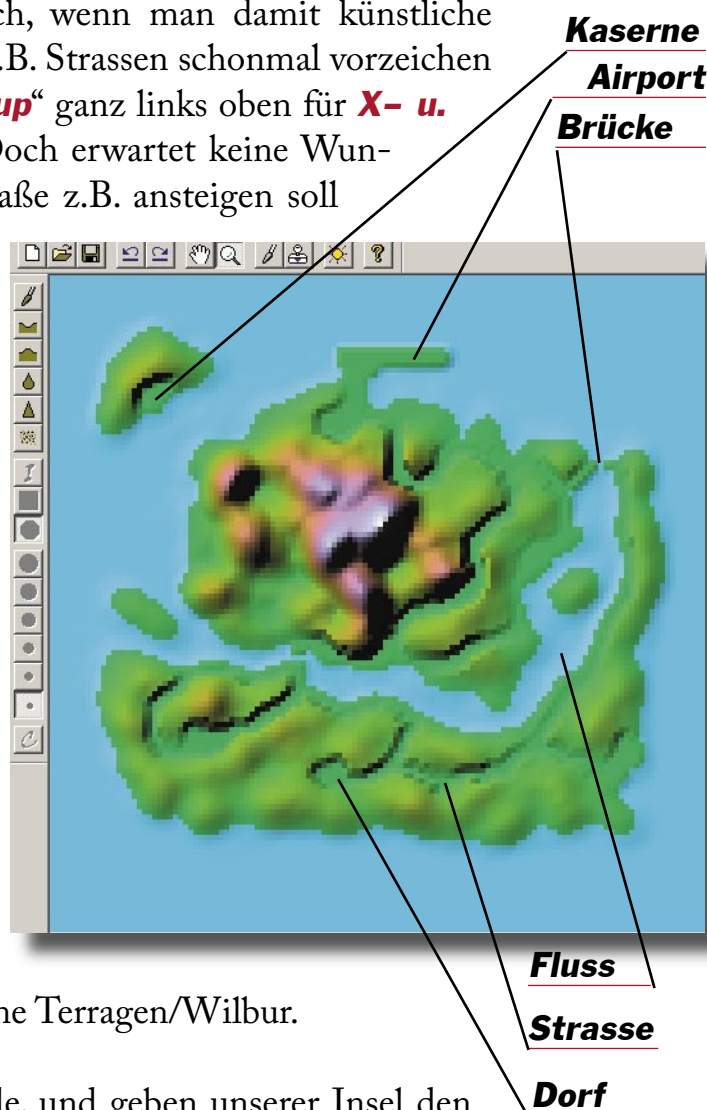
Vorher



Nachher

„**Lerp**“ hingegen (ganz wichtige Einstellung!) bedeutet, dass nun der Wert „**Value**“ eine konstante Höhe bedeutet, und diese immer dort gezeichnet wird, wo der Pinsel eingesetzt wird. Ein Beispiel: Stellt erst mal euren Pinsel auf „**Add**“, und „**Value**“ auf „10“ ein – dann mit „**OK**“ bestätigen. Malt nun auf der Map ein wenig herum – ganz egal, wie es genau aussieht. Hauptsache, die Insel erhält erstmal „**Fleisch**“. Momentan sieht das Ganze noch ein wenig Fremdartig aus, aber das haben wir gleich. Ganz oben seht ihr eine **kleine, gelbe Sonne**. Klickt einmal darauf, und schon sieht eure Insel wie eine Insel aus (siehe kleines Beispiel links). Jetzt stellen wir den Pinsel im **Brush-Setup** auf „**Lerp**“ um, und geben bei „**Value**“ z.B. „5“ ein. Wieder mit „**OK**“ bestätigen, und erneut auf der Map malen. Seht ihr den Unterschied? So könnt ihr einen Canyon ins Gebirge fräsen, einen Flughafen ins Meer bauen, oder Strassen und Dorfplätze planieren. **Beachten** sollte man dabei nur eins: OFP hat **Gezeiten!** Und die höchsten Tidenhübe gehen bis +5 Meter über Normal Null. D.h. für uns: wenn wir nasse Füße vermeiden wollen, sollten wir den „**Value**“ nicht allzu niedrig wählen. Wieviel genau, hängt davon ab, auf

welche maximalen Höhenwerte wir später unsere Insel spreizen werden, bevor wir sie abspeichern. Als Daumenregel könnte folgendes gelten: Wer seine Insel später auf die doppelte Höhe spreizt, braucht einen Mindest-Value von ~ 3 . Wer an der Höhe nichts ändern möchte, sollte ~ 6 als Value eintragen. Ein weiterer nützlicher Verwendungszweck der Funktion „**Lerp**“ ist das Anlegen von passierbaren Übergängen in Flüssen. Allerdings ist hierbei eben zu berücksichtigen, auf welchen Breitengrad die Insel in OFP eingestellt wird, und in welcher Tageszeit die Mission dann spielt. Experimentieren ist an dieser Stelle angesagt. Zurück zu unserer Insel: Spielt zunächst einfach mal mit den verschiedenen Einstellungen des Pinsels herum, und verändert auch mal die Größen des Werkzeugs. Das geht ganz einfach in der Palette links: einfach auf die verschiedenen **grünen Kreise** klicken, und schon habt ihr mal große, mal kleine Pinsel zur Verfügung. Wer möchte, kann dort auch den Pinsel von **Rund** auf **Quadratisch** umstellen. Sicher ganz nützlich, wenn man damit künstliche Strukturen schaffen will. Wer z.B. Strassen schonmal vorzeichnen möchte, sollte im „**Brush-Setup**“ ganz links oben für **X- u. Y-Werte** eine „3“ eingeben. Doch erwartet keine Wunder dabei, denn sobald die Straße z.B. ansteigen soll (wie ein Pass), kann es sehr, sehr mühselig werden. Aber man kann z.B. das Bauen von Brücken recht gut vorbereiten, oder das Planieren von Kasernenhöfen. Rechts ein Bild, wie so eine Insel z.B. aussehen könnte. Behalten wir einfach folgendes im Gedächtnis: Wer sich in Wilbur leicht zurechtfindet, kann hier schon recht gute Ergebnisse erzielen. Wer sich jedoch eher schwer in dieser 2D-Umgebung tut, der hat immer noch in WrpTool die Gelegenheit, seinen Vorstellungen Gestalt zu verleihen. Notfalls sogar ganz und gar ohne Terragen/Wilbur.



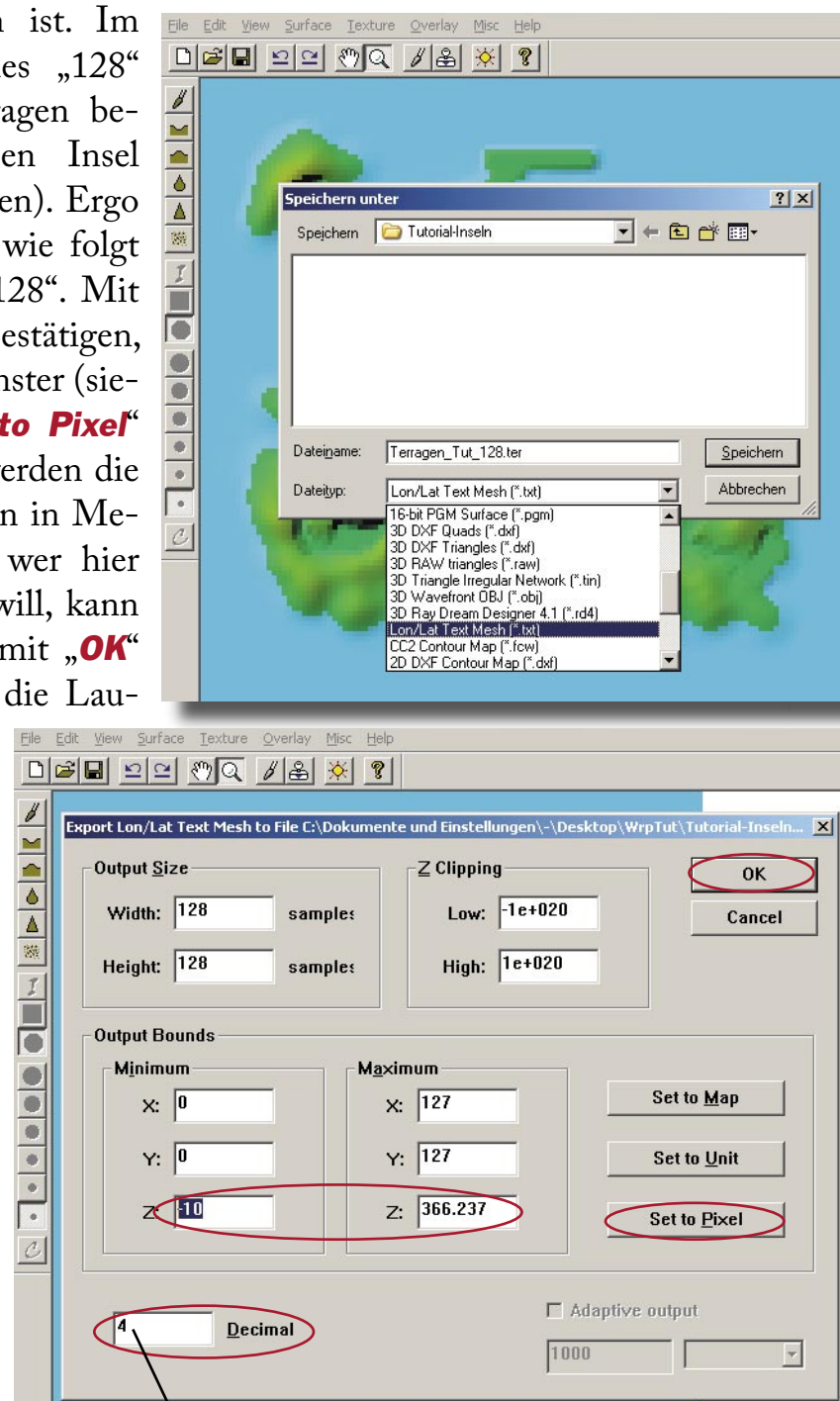
Kommen wir nun zum Ende, und geben unserer Insel den letzten Schliff, indem wir unsere Vorstellungen bezüglich der endgültigen **Höhe/Meerestiefe** umsetzen. Wie Hoch/Tief unsere Insel momentan in Metern ist, lässt sich leicht überprüfen: rechts unten zeigt Wilbur die aktuelle **Mausposition**, sowie

an letzter Stelle die **Höhe in Metern** an. Wem seine Insel bereits hoch genug ist, kann jetzt sofort zu „**Speichern**“. Wer sich gerne in höheren Regionen tummelt, dem wird jetzt geholfen: Öffnet das Menü „**Surface**“ > „**Point Process**“ > „**Scale**“, und gebt eure Höhen-/Tiefenvorstellungen dort in das Dialogfeld ein. Doch erinnern wir uns stets daran, dass WrpTool ab 1474m zu clippen beginnt. D.h. dass dann die Berggipfel quasi den Meeresboden erreichen! ;-) Außerdem wirken fast 1500 Meter in Meeresnähe ausreichend respektgebietend und eindrucksvoll.

So weit, so gut. Speichern wir nun unsere Insel ab. Dazu müssen wir im **Speichern-Dialog** (**File** > **Save as**) das Dateiformat „**Lon/Lat Text Mesh (*.txt)**“ auswählen (siehe Bild). Wir geben der Map noch einen Namen, in dem möglichst die Inselgröße enthalten ist. Im Beispielfall wäre dies „128“ (wer jedoch in Terragen begonnen hatte, dessen Insel müsste „256“ enthalten). Ergo könnte unsere Insel wie folgt heißen: „MyWorld_128“. Mit Speichern dann bestätigen, und im folgenden Fenster (siehe unten) auf „**Set to Pixel**“ klicken. Daraufhin werden die aktuellen Höhendaten in Metern angezeigt, und wer hier nochmal eingreifen will, kann dies jetzt tun. Jetzt mit „**OK**“ bestätigen, fertig ist die Laube. **Wichtig: Lasst Wilbur noch offen! Bitte erst schließen, wenn ihr mit dem Ergebnis zufrieden seid!**

Wichtig: Lasst Wilbur noch offen! Bitte erst schließen, wenn ihr mit dem Ergebnis zufrieden seid!

Fassen wir zusammen: Wir haben jetzt eine Rohdatei von einer Insel erstellt, die schon mehr oder weniger unseren Vorstellungen entspricht. Einmal ha-



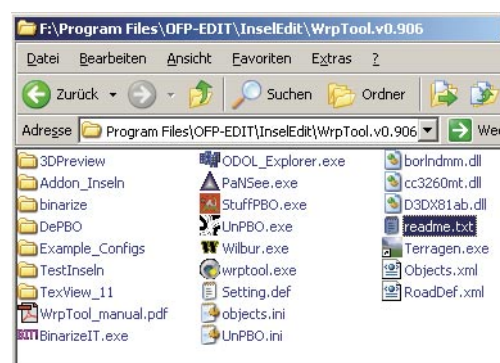
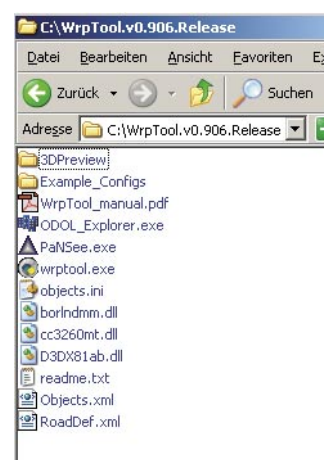
„0“ eintragen

ben wir ein besonders natürlich wirkendes Terrain mit Terragen erstellt, und alternativ dazu eine eher „maßgeschneiderte“ Version direkt in Wilbur gestaltet. Bleibt noch zu sagen, dass es noch weitere Möglichkeiten, insbesondere **Kombinationsmöglichkeiten** beider Progs gibt. Wer möchte, kann z.B. in Terragen eine Insel erstellen, und diese in Wilbur auf eine **kleinere Größe herunterrechnen** lassen. Größer geht zwar auch, aber die Qualität ließe dabei arg zu Wünschen übrig. Ferner kann man natürlich – wie schon erwähnt – Terragen-Inseln in Wilbur nachträglich noch weiterbearbeiten, wie dies schon beschrieben wurde. Bleibt noch zu sagen, dass es wenig Sinn macht, nachher in WT eine Insel zu wrappen, mit deren Terrainstruktur man noch nicht richtig zufrieden ist. Man sollte seine Experimente und Versuche erst dann einstellen, wenn das Ergebnis wirklich zufriedenstellend ist, bzw. sich seine Vorstellungen mit bestehendem Wissen nicht besser umsetzen lassen. Doch wie beurteile ich nun, ob meine Insel meinen Vorstellungen entspricht, in welchen Bereichen noch Nachbesserungen erforderlich wären? Ganz einfach: in **OFP** natürlich. Hä? Schon richtig, noch wissen wir ja nicht, wie das gehen soll. Aber keine Panik, das kommt schon noch. Als nächstes werden wir WrpTool installieren, und so zügig wie möglich an einen Punkt zu gelangen, der es euch ermöglicht, eure – vorerst noch fast „nackten“ – Inseln schonmal in OFP betrachten zu können. Ist dies geschehen, werde ich darauf hinweisen, dass nun der geeignete Zeitpunkt gekommen ist, seiner Insel terrainmäßig den letzten Schliff zu geben, bzw. notfalls auch nochmal neu zu beginnen. Denn nichts ist frustrierender, als nach stundenlanger Arbeit festzustellen, dass man ein **„totes Pferd geritten“** hat....



WrpTool

OK, es ist soweit – letzte Gelegenheit für ein oder zwei Mutpillen, denn wir **installieren** jetzt **WrpTool**! Nein nein, so schlimm ist es nicht, und wenn ihr euch genau an meine Anweisung haltet, dann „funzt“ auch alles. Es ist halt wichtig, diese Schritte gewissenhaft durchzuführen, da bei der Installation die meisten Fehler passieren (So jedenfalls die Aussage der Entwickler). Ich gehe mal davon aus, dass ihr die WrpTool.rar-Datei schon entpackt habt, und euch jetzt der entpackte Ordner „**WrpTool.v0.950.Release**“ irgendwo vorliegt – prima. In diesem Ordner liegt ebenfalls einer mit dem gleichen Namen. Diesen 2. Ordner gleichen Namens verschieben, oder noch besser, kopierem wir nun an eine definierte Position auf eurer Platte. Prinzipiell kann das überall sein, solange ihr genau wisst, was ihr tut. Doch für den Anfang, und für unser Beispiel wählen wir jetzt **C:** als Speicherort. Unser Pfad lautet momentan also: **C:\WrpTool.v0.950.Release\...** Soweit, so gut. Öffnen wir doch mal diesen Ordner, und werfen einen Blick hinein: Wir finden dort das Hauptprogramm **wrptool.exe**, sowie **ODOL_explorer.exe**, **PaNSee.exe**, ect... siehe kleines Bild rechts. Wer mag, kann in diesen Ordner nun alle weiteren benötigten Tools hineinkopieren, sowie noch einen Ordner für Testinseln, Rohdaten, und Insel-Addons von anderen Wrappern/Mods anlegen. Das Ganze könnte dann wie im 2. Bild aussehen. Aber wiegesagt: nur, wer mag. Genausogut geht auch ein externer Ordner, oder ihr lasst die Tools dort, wo ihr sie immer aufbewahrt, falls schon installiert. Hauptsache, ihr kennt euch dabei aus. Und genau das ist auch der Punkt, der für WrpTool so **wichtig** ist: es muss sich ebenfalls „auskennen“, sprich, wo es seine benötigten Dateien finden kann. Welche Dateien das sind? Zunächst sind dies alle **Insel-Textures**, sowie sämtliche **xxx.p3d-Files** der verwendeten **Objekte**, und deren **Textures**. Diese Daten sind alle in **OFF/OFF-RES** enthalten, wir müssen sie nur noch aufbereiten. Zunächst legen wir jedoch den zentralen, und alles entscheidenden Ordner für diese Daten an. Wie ihr ihn nennt, ist letztlich egal. Ich habe ihn für das Beispiel „**DePBO**“ genannt, womit schonmal angedeutet wird, was sich darin befindet: nämlich **entpackte** Daten, und er sollte sich im Hauptordner „**WrpTool...**“ befinden. Nun öffnen wir „**UnPBO.exe**“, und entpacken folgende



Dateien aus **OFFP / DTA** in den soeben erstellten Ordner:

Abel.pbo

Cain.pbo

Data3D.pbo

Data.pbo

Eden.pbo

Fonts.pbo

LandText.pbo

und aus **OFFP\Res\Dta**-Ordner folgende Dateien (wenn Meldung erscheint, einfach dem Ersetzen der Dateien zustimmen!):

Data3D.pbo

Data.pbo

Fonts.pbo

LandText.pbo

sowie aus **OFFP\Res\Addon**-Ordner:

O.pbo

Noe.pbo kommt auch hierher, wird später zum Testen benötigt.

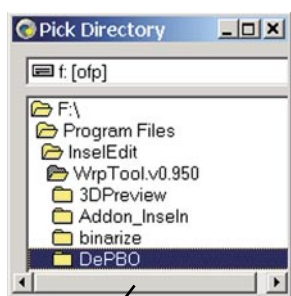
Später, wenn alles klappt, und wir uns in WrpTool gut zurechtfinden, können wir natürlich auch die inoffiziellen Addons verwenden, indem wir die jeweiligen Inseln/Objekte ebenfalls in diesen Ordner entpacken (siehe auch Manual). Doch denken wir stets daran: Sobald irgendein Fitzelchen **Inoffizielles** verwendet wurde, oder Originales verändert wurde, muss dies für jeden glasklar erkennbar und **ausgewiesen** sein (so wie es auch beim Scripten und bei Mods schon lange die Regel ist)! Es geht also nicht an, sich z.B. Novoga in WrpTool zu laden, dort ein paar scheinbar „unbedeutende Anpassungen“ durchzuführen, und hernach das Teil wieder unter „Noe.wrp“ abzuspeichern.

BÖSEBÖSEBÖSEDAS!

Der eine oder andere mag sich vorhin gewundert haben, dass wir manche Dateien quasi doppelt entpackt hatten. Doch das macht schon Sinn, da nun im Ordner „Data3D, Data, Fonts, ect. die Dateien von BEIDEN Versionen enthalten sind. Und noch manch Anderer mag sich – allerdings erst später – wundern, wenn etwas nicht funktioniert, falls er sich nicht an meine Anweisung

gehalten hat, und mit einem anderen Tool als **PBO Decryptor v1.5 (=UnPBO.exe)** entpackt haben sollte. Denn NUR mit diesem Tool funzt alles, wie es soll! Glaubt es mir, ich hab den „Test“ schon hinter mir! ;-) Und noch was wichtiges: Lasse sich keiner einfallen, die Dateistrukturen, die der Entpacker angelegt hatte, zu ändern – das wäre der Anfang von Ende! Einfach in den „DePBO“ entpacken und ggf. überschreiben lassen. Aber **nie, nie, nie** dort drin was ändern! Es gibt nur zwei Ausnahmen: Wer im Laufe der Zeit mit Wrappen fit geworden ist, und bisher alles geklappt hat, kann sich – wenn er will – die (unnötige) Mühe machen, und sämtliche unnötigen Files entfernen. Es befinden sich ja nicht nur statische Objekte, sondern auch eine Menge dynamische Objekte im Ordner, die nie benötigt werden. Zweitens kann man (später!) natürlich noch die inoffiziellen Addons hierher entpacken.

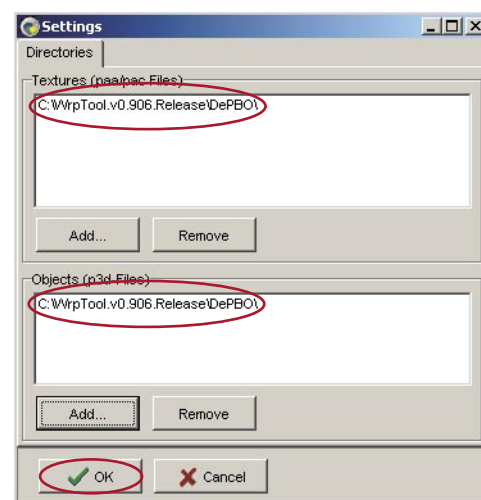
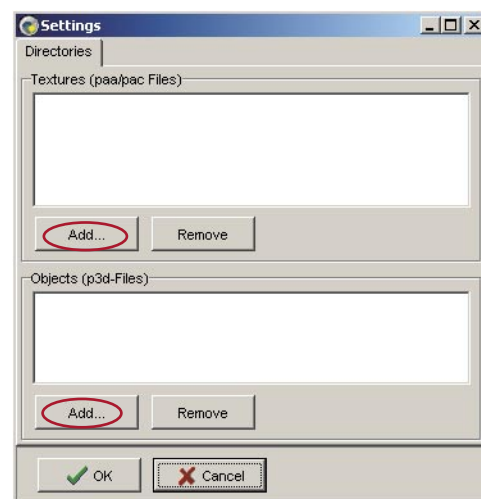
So weit, so gut. Teilen wir WrpTool also nun mit, wo es sich seine Dateien holen kann. Dazu öffnen wir WrpTool zum ersten mal und freuen uns über die aufgeräumte, durchdachte und übersichtliche Oberfläche :-). Wenn wir nun oben in der Menüleiste auf „**Settings**“ > „**Change Settings**“ klicken, erscheint ein Fenster (rechts), indem wir per „**Add**“ den Pfad (Bild links) zu unserem „**DePBO**“-Ordner einstellen. Hier muss mit „**Enter**“ bestätigt werden – ein Button hierfür wird vergeblich



Bestätigung
mit „Enter“

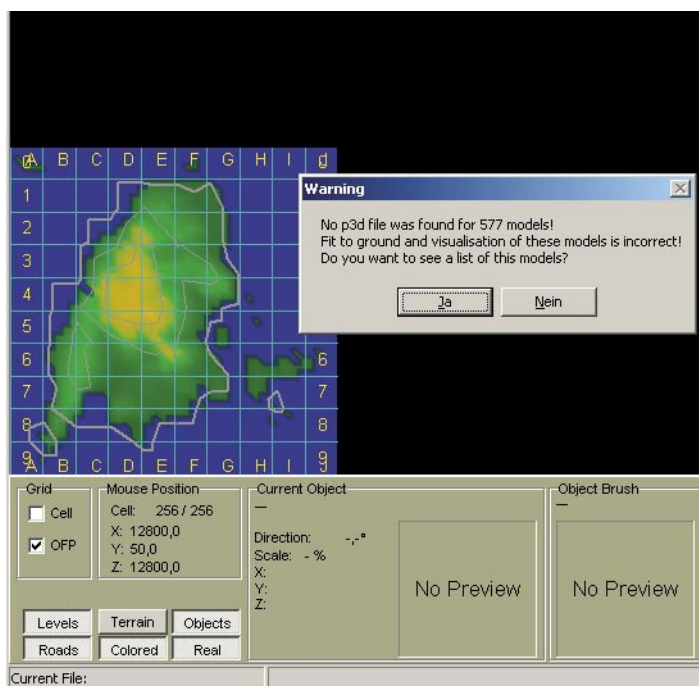
gesucht. Das machen wir natürlich in BEIDEN Feldern (Textures und Objects). Bild 2 rechts zeigt, wie das dann aussehen sollte. Wenn dies geschehen ist, bestätigen wir mit „**OK**“, und starten WT sicherheitshalber nochmal neu. Der eine oder andere mag nun vermuten, dass sich hier auch mehrerer Ordner und Pfade eingeben lassen. Und richtig, das ist auch der Fall. Ich persönlich operiere hier mit mehreren Einträgen, um den Überblick beim Testen zu behalten. Wer seine erste Installation erfolgreich

bewerkstelligen konnte, kann sich später gerne eine neue Konfiguration basteln. Doch für Anfänger wäre dies sicher nur verwirrend, weswegen wir uns hier ausschließlich mit der von den Entwicklern empfohlenen Konfiguration befassen, und diese exakt wie beschrieben umsetzen. Alles andere geschieht auf eigenes Risiko, und wer dann flennt, weil's net funzt, wird erschossen! ;-)



Vorbereitungen

Um unsere soeben abgeschlossene Installation prüfen zu können, werden wir einen Test durchführen. Im „**DePBO**“ > „**Noe**“-Ordner sollte sich die Datei „**Noe.wrp**“ befinden. Gefunden? Gut. Öffnet diese Datei mit WT („**File**“ > „**Load WRP...**“), und habt ein wenig Geduld....Festplatte:*rödelrödelrödel*...OK. Wenn JETZT KEINE Fehlermeldung kommt, und WT eine Insel (nämlich Novoga) anzeigt, dann haben wir schon so gut wie gewonnen. WT sollte nun ganz rechts unten im Aktionsfenster als letzten Eintrag „**Load road system**“ anzeigen. Ja? Fein. Sollte jedoch bei der Installation etwas schief gegangen sein, könnte die Situation sich etwa folgendermaßen darstellen: WT lädt und röhlt zwar heftigst, meldet sich aber plötzlich mit einer Fehlermeldung, in der irgendetwas von fehlenden Objekten oder so genuschelt wird, und ob man diese angezeigt haben möchte. Sagen wir erstmal „NEIN“, und betrachten, was geschehen ist. Wir haben zwar jetzt auch eine Insel, jedoch werden die Objekte angezeigt, ohne deswegen tatsächlich vorhanden zu sein. Komme also keiner auf die Idee, es sei alles in Ordnung, weil man doch Straßen, Häuser und sonstwas auf der Map erkennen kann (zumindest, wenn man ein wenig mit dem Mause rad hineingezoomt hat). Das ist ein fürchterlicher Trugschluss, der folgendermaßen zustande kommt: WT arbeitet, wie auch OFP selbst, nur mit sogenannten „**Zeigern**“, die in den xxx.wrp-Files enthalten sind. Diese Zeiger enthalten Informationen über Art, Position, sowie Lage eines Objekts, und wo sich die Objektdatei tatsächlich befindet. Die eigentlichen Objekte liegen also ganz woanders – nämlich genau dort, wo wir unseren Ordner „DePBO“ erstellt hatten, und mitnichten in der eben geladenen „Noe.wrp“. Die Map in WT braucht jedoch keine Objektdaten, sondern nur die Info aus „Noe.wrp“, welches Objekt sich wo, und in welcher Lage befindet. Sogar die Vorschau auf ein Objekt funktioniert tadellos, da die Zeigerstrukturen ja auf eine tatsächlich vorhandene Bilddatei (im Ordner „3DPreview“) ver-



weisen. Deswegen können wir problemlos eine scheinbar intakte Insel betrachten, und könnten auch munter drauflos editieren. Hilfe – bloß nicht! Einfacher Beweis, dass das in die Katastrophe führt: Unter „**View**“ > „**3D-View**“ öffnet ihr das 3D-Vorschaufenster. Normalerweise müssten nun eine Menge Objekte darin zu sehen sein (nahe genug im Hauptfenster mit Mausrad reinzoomen!). Doch da sind keine zu sehen, obwohl doch im Hauptfenster alles voll damit ist! Das liegt einfach daran, dass 3D-View in genau dem Ordner nach den tatsächlichen Objekt-Files sucht, den wir in den Settings angegeben hatten. Wenn sich also dort keine Daten befinden, oder diese verändert wurden, wird's nix mit dem Nachbarn, und 3D-View ist beleidigt ;-)

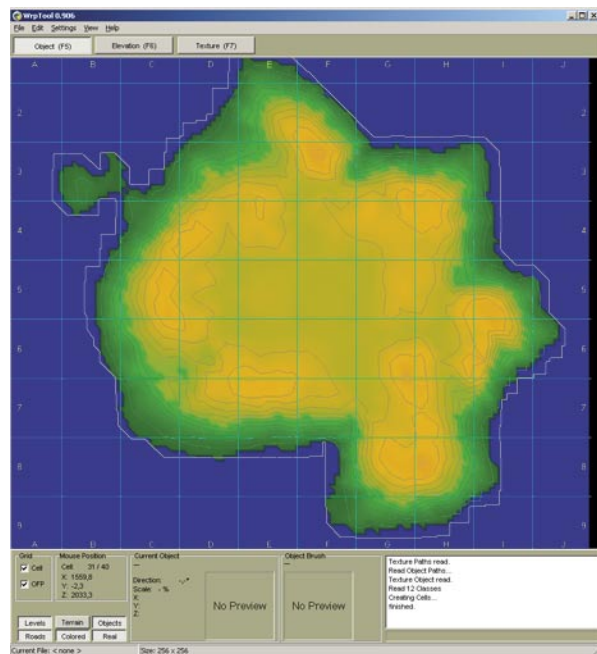
Ein weiterer Hinweis auf Fehler sind übrigens die blauen Felder. Zusammenfassend kann also gesagt werden: Zeigt 3DView garkeine, oder auch nur bestimmte Objekte NICHT an, braucht man garnicht erst weitermachen, sondern muss erst das Problem beseitigen. Ich habe mich speziell darüber extra ausführlich ausgelassen, weil hier Fehler einfach jede Arbeit zunichte machen, wenn man sie JETZT nicht bemerkt. Ich hoffe, ich konnte ein wenig Verständnis für die inneren Strukturen von WT/OFP vermitteln. Ich denke, das kann nie schaden, oder?

Denn ihr wisst ja jetzt, was sonst wieder passieren kann: ... ;-)



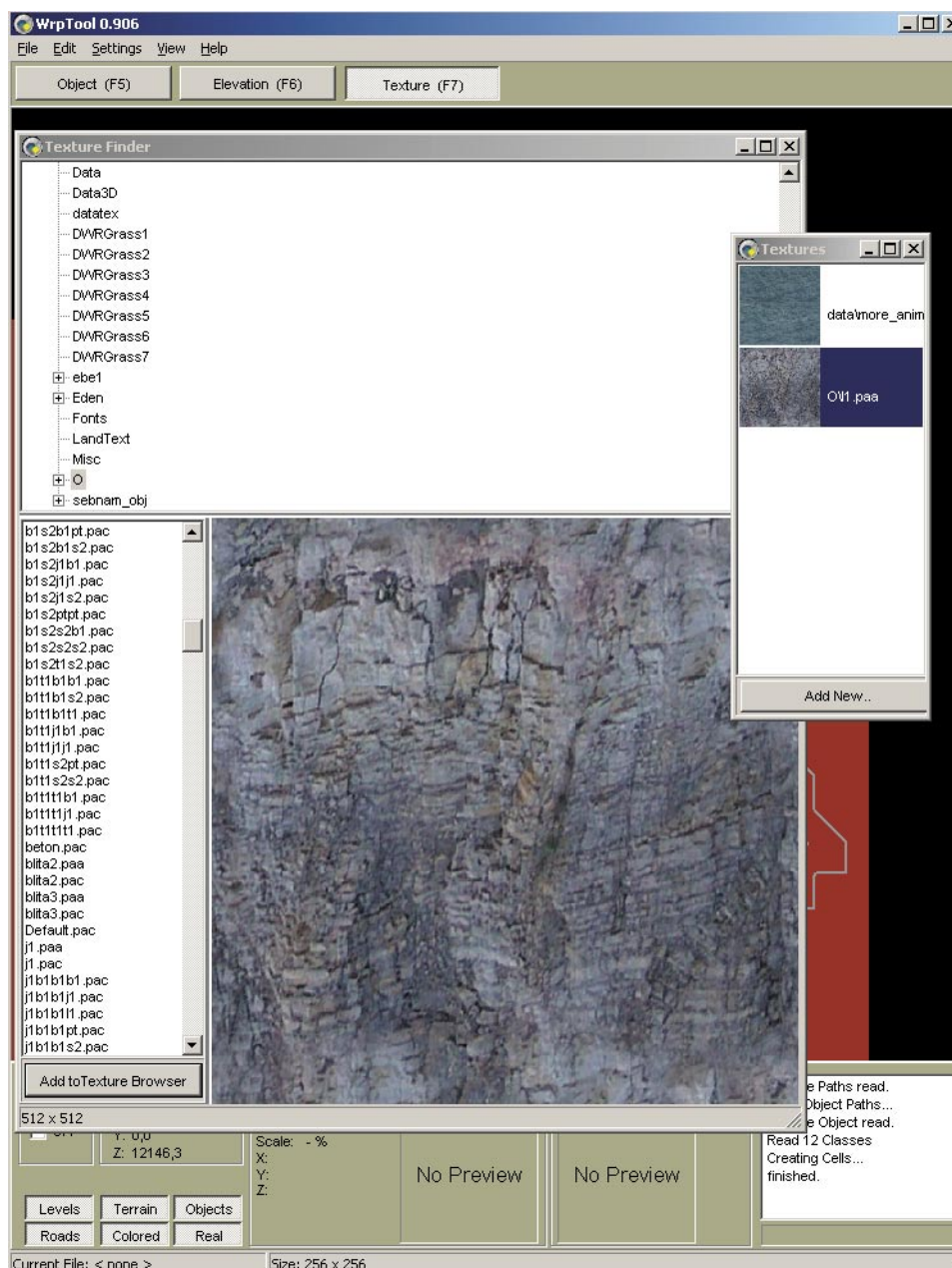
Allmählich gehts ans Eingemachte, und wir laden zum ersten Mal (endlich) unsere erste – vorhin in Wilbur erstellte – Inseldatei in WrpTool. Wenn mich nicht alles täuscht, müsste die Datei **„MyIsland_128.txt“**, oder für „Terragenisten“ ..._256.txt heißen, richtig? Gut. Klicken wir in WrpTool zuerst auf **„File“** > **„New“**, und geben im Dialogfenster bei **„Map Size“** ebendiese „128“, oder „256“ (Terragenisten) ein. Nun wird auch klar, warum wir unsere Inselbezeichnung mit diesem Anhängsel versehen hatten – denn nach ein paar Tagen, oder bei vielen Inseln weiß man das bald nicht mehr auswendig. Also weiter. Nachdem nun im Hauptfenster ein mehr oder weniger großes, blaues Feld erschienen ist, klicken wir erneut auf **„File“**, und diesmal auf **„Import Wilbur“**. Hey, das wars! War doch nicht schwer, oder? :-)

Unsere Insel sieht jetzt schonmal richtig gut aus, und wenn wir ihr noch schnell eine Grundtextur verpassen, können wir unsere „Erstgeburt“ auch schon in OFP bewundern. Losgeh:



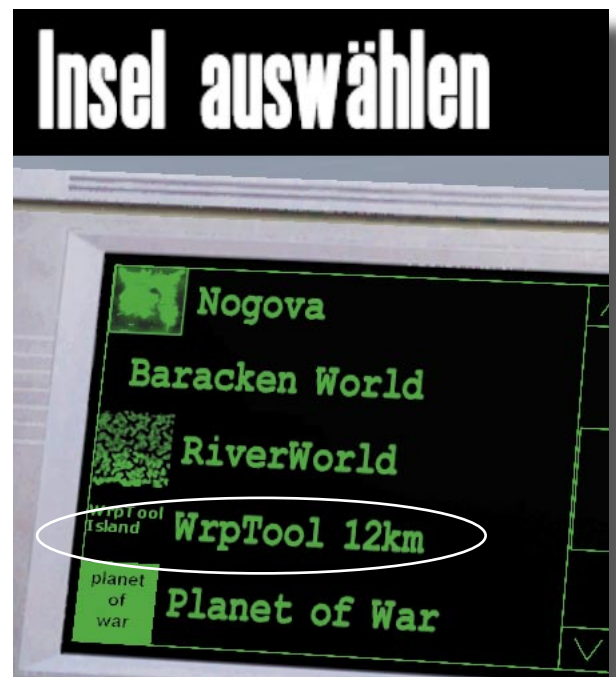
Wir wechseln nun oberhalb des Hauptfensters in den **Textur-Modus (F7)**, und ziehen mit der Maus einen Rahmen über die gesamte Map – nun sollten alle Zellen markiert sein, und das ganze etwa wie rechts gezeigt aussehen. Öffnen wir als nächstes den **„Texture-Browser“** im **„File“-Menü**, und wundern uns ein bisschen. Nur eine Texturkachel? Ist aber ganz normal so. Zeigen wir WT also, wo noch mehr zu holen ist: Klickt unten im **„Texture-Browser“** auf **„Add New“**, und dann im nun geöffneten **„Texture-Finder“** auf das **„+“-Zeichen ganz links oben**. Nun sollte der Pfad aufklappen, und ein ganzer Kürbis voll neuer Einträge müsste zu sehen sein (Bild nächste Seite). Sucht nach dem Eintrag **„O“** (steht für Resistance-Dateien), um dort nach einer Textur zu fahnden. Dazu brauchen wir „O“ nicht aufklappen, es reicht „O“ anzuklicken, schon sind um unteren Fenster haufenweise Texturen aufgelistet. Sobald nun eine davon angeklickt wird, erscheint rechts daneben eine Vorschau. Entscheiden wir uns zunächst für **„l1.paa“**, klicken sie an, und dann auf **„Add to Texture-Browser“**. Voilà, schon haben wir eine brauchbare Textur, um unsere Insel zu wrappen. Dazu klicken wir





einmal im „**Texture-Browser**“ auf die gewünschte Textur, damit diese angewählt ist, und halten dann die „**Shift**“-Taste, während wir mit der Maus irgendwo auf die Map im Hauptfenster klicken. Bingo, unsere Insel ist jetzt texturiert, und damit bereit, in OFP geladen zu werden. Also gehen wir wieder ins „**File**“-Menü und speichern die Insel mit „**Save WRP**“ in OFP/Worlds. Als Dateiname vergeben „**MyWorld_128**“ (oder eben ..._256). Anschließend kopieren wir aus dem mitgelieferten „**Example-Configs**“-Ordner folgende Dateien in einen neuen Ordner, der genauso wie unsere Insel benannt wird: „**config.cpp**“ und „**wrptool_12km.paa**“. In der **config.cpp** tauschen wir alle „**wrptool_12km**“-Einträge in „**MyWorld_xxx**“ um (die „xxx“ stehen für eure Inselgröße). Nur den Eintrag bei „**cutscenes**“ und „**icon**“ lassen wir unberührt. Speichern und schließen wir die Config.cpp nun, und rufen einen **PBO-Packer** auf, mit dem wir dann diesen Ordner zu einer PBO machen. Zum Schluss schieben wir diesen, sowie

den **“wrptool_12km_anims“** aus **„Example_Configs\OFP\Addons“** in unseren OFP/Res/Addons-Ordner. Jetzt müsste im OFP-Editor eine neue Insel vorhanden sein, die mit **„WrpTool Island“** anstatt eines Bildes bezeichnet ist - das ist eure Insel. Wie ihr euer eigenes Bild erstellt, erfahrt ihr gegen Ende des Tuts noch. Jedenfalls wählt ihr diese Insel nun aus, und im Editor erstellt ihr am besten gleich einen HELI „fliegend“ auf der Map. Vorschau klicken, und voilà, choppert ihr über eure **erste eigene Insel!** War doch nicht schlimm bis jetzt, oder? ;-)

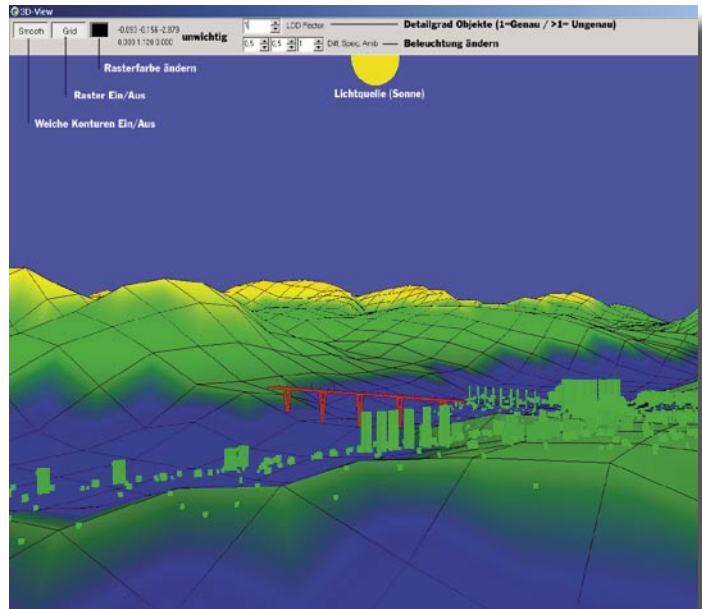


Nun haben wir es also tatsächlich geschafft, ein erstes Erfolgs-Erlebnis zu erzielen! Und genau hier ist auch der Punkt, an dem wir nun – wie schon angekündigt – unsere Insel mit dem Heli genauer untersuchen sollten, ob das Terrain in allen Punkten zu unserer Zufriedenheit gelungen ist. Denn wenn nicht, macht ein **Weiterarbeiten keinen Sinn**. Ergo kehren wir immer an dieser Stelle entweder wieder zu unseren vorherigen Programmen zurück um nachzubessern, oder gleich nochmal ganz von Vorne zu beginnen. Wer in Wilbur begonnen hatte, und dieses mittlerweile – entgegen meinem Rat ;-)) – geschlossen haben sollte, hat sowieso (fast) keine andere Wahl, als nochmal neu zu beginnen. Terragenisten könnten immerhin noch ihr „**xxx.ter**“-File in Wilbur laden, und dort die Insel nachbessern. Aber auch für die „Wilburianer“ gibt es noch Hoffnung. Denn generell gilt: Auch WrpTool ist in der Lage, das Terrain zu gestalten. Notfalls – wie schon erwähnt – sogar ganz ohne Zusatzprogramme – **Cool!** Machen wir es deshalb so: Betreten wir nochmal Neuland in WrpTool, bevor wir ggf. zu den vorherigen Programmen zurückkehren. So kann dann ein Jeder selbst beurteilen, wie er ggf. seiner Insel den letzten Schliff verpassen möchte – Einverstanden?

3D-View+Mini-Map

Damit wir uns beim Anpassen unserer Insel gut zurechtfinden können, besprechen wir vorher kurz die Mini-Map, und die 3D-View. Beide finden wir im Menü „View“. Zur Mini-Map sollte eigentlich alles klar sein – sie birgt keine Geheimnisse. Einfach auf die gewünschte Region auf der Mini-Map klicken, und schon flippt die Map auf den gewünschten Bereich. 3D-View ist auch schnell erklärt: Man sollte nur immer daran denken, dass 3D-View nicht grenzenlos Zoomen kann, ohne dass manchmal auch im Hauptfenster „nachgezogen“ wird. Rotiert wird einfach mit **LMB** (**L**inker**M**aus**B**utton), verschoben mit **RMB**, und gezoomt mit „**Strg**+**LMB**“. Ansonsten muss man wissen (aber man merkt es auch so), dass 3D-View nicht über den im Hauptfenster angezeigten Bereich „hinaussehen“ kann. Somit hat man es fast immer mit einer Kombination von Hauptfenster und 3D-View beim Betrachten zu tun. Dafür ist 3D-View aber ein „Echtzeit-Viewer“, bei dem nicht ständig aktualisiert werden muss. Jedes Hinzufügen, Entfernen, Verschieben von Objekten wird **SOFORT** angezeigt.

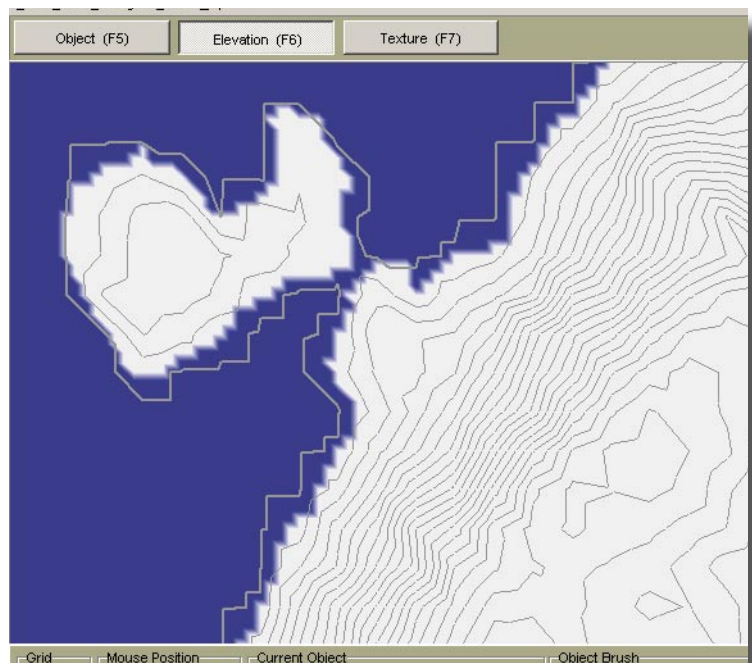
Ebenso, wenn sich an der Terraingeometrie was ändert – also genau das, was wir jetzt brauchen. Die neue Version von 3D-View hat noch ein paar Features im oberen Fensterbereich spendiert bekommen. Im Grunde kann man alles so lassen, wie es ist. Wer möchte, kann die Zellendarstellung An- u. Abschalten, oder die Darstellung von geglättet auf „kantig“ umschalten. Auch die Beleuchtungsverhältnisse lassen sich nun beeinflussen. Neu ist auch, dass nun die unterschiedlichen Höhenverhältnisse farblich differenziert dargestellt werden.



Elevation-Mode

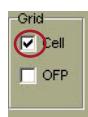
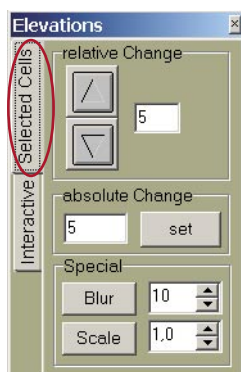
Wir haben also vorhin festgestellt, dass unsere Insel noch nicht unseren Vorstellungen entspricht, und wollen das nun direkt hier in WrpTool ändern. Auf meiner Beispielsinsel hätte ich gerne ein bebauungsfähiges Gelände für Dorf, Straßen, Brücke und einen Flughafen auf der Insel. Das ist das Ziel, und wir lernen jetzt, wie das geht. Solltet Ihr WT vorhin geschlossen haben, bitte neu starten, und die Übungs-Insel wieder laden.

Um jetzt in den Elevations-Mode zu kommen, drücken wir den mittleren Button oben, oder „**F6**“. Was wir nun vor allem brauchen, sind die Höhenlinien auf der Map. Wenn diese nicht schon zu sehen sind, klicken wir links unten auf „**Levels**“. Alle anderen Ansicht-Buttons schalten wir ab, damit wir einen guten Kontrast erzielen. Die Mapdarstellung sollte dann ungefähr wie rechts aussehen.



Elevations-Tool

Das Elevation-Tool finden wir (wie fast alles) unter dem Menü „**View**“, oder mit „**Strg**+“**E**“. Betrachtet nun das Fenster (links).

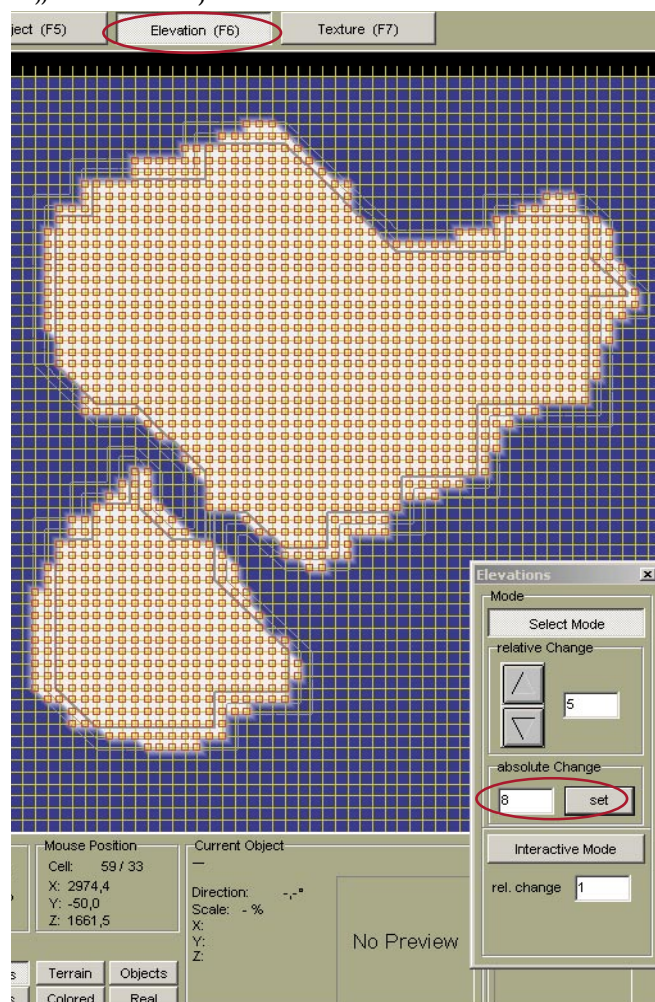


Es sollte immer die „Cell“-Einstellung aktiv sein!

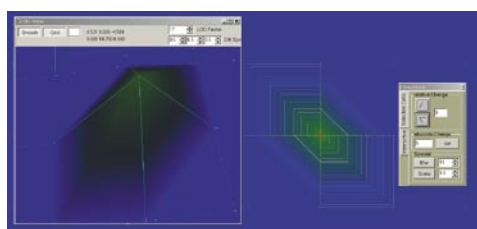
Links befindet sich der Button für den „**Select-Mode**“. Das bedeutet, dass in diesem Modus WT immer mitgeteilt werden muss, welche Zellen manipuliert werden sollen. Das geschieht recht einfach, indem wir eine oder mehrere Zellen mit der Maus selektieren. Wobei der Ausdruck „Zellen“ in diesem Modus nicht ganz passend ist. Es sind die Schnittpunkte der Zellenlinien, die sich manipulieren lassen. Aber der Einfachheit halber bleibe ich bei der Bezeichnung „Zellen“ – gemeint sind dabei jedoch die Schnittpunkte. Lasst uns doch einfach mal – damit wir ein Gefühl dafür bekommen, was das Elevation-Tool so drauf hat – eine

leere Map erstellen (**File**>**New**>Mapsize **64**>**OK**). Ein Blick auf das Koordinatenfenster links unten verrät uns, dass WT automatisch eine Fläche 50 Meter unter dem Meeresspiegel erstellt hat. Aktivieren wir nun – falls nicht schon geschehen – links unten das Häkchen bei „**Cell**“ im „**Grid**“-Bereich. Markieren wir jetzt einen beliebigen Bereich auf der Map. Im Eingabefeld bei „absolute Change“ tragen wir eine 8 ein, und klicken auf „Set“. Voilà, schon haben wir

eine Insel – wenn sie auch noch flach wie eine Flunder ist. Doch zunächst aktivieren wir mal unser 3D-View-Fenster (Menü „**View**“ > „**3D-View**“), um eine bessere Kontrolle über unsere Arbeit zu ermöglichen. Die komischen „Franzen“, die am Map-Rand zu sehen sind, sollen uns nicht stören, man könnte ohnehin nichts dagegen tun. Als nächstes wollen wir erneut ein paar Zellen markieren, doch diesmal deutlich innerhalb der Inselgrenzen, um das nächste Tool zu testen. Klickt diesmal auf den „**Pfeil-nach-oben-Button**“, und beobachtet im 3D-Fenster, was sich tut. Soweit alles klar? War ja nicht schwer. Störend dabei ist halt, dass die Insel immer noch furchtbar künstlich aussieht. Aber auch dem kann abgeholfen werden



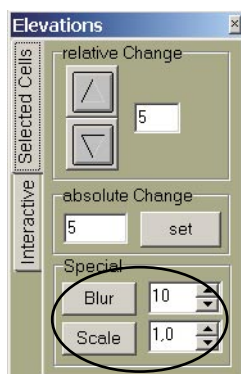
– mit dem letzten Tool. Dazu müssen wir den „Select-Mode“ verlassen, indem wir auf den „**Interactive Mode**“-Button klicken. Jetzt können wir mit der Maus unser Inselterrain wie mit einem Pinsel



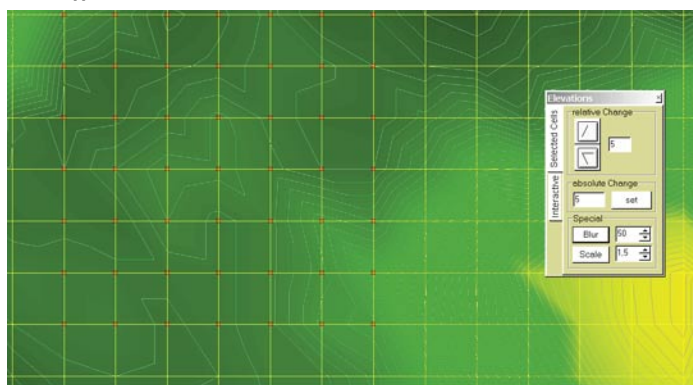
„organischer malen“. Bei gedrückter „**Shift**“-Taste wird der Boden gesenkt. Im Bild links seht ihr, wie WT die Map beeinflusst, wenn nur ein Schnittpunkt markiert ist. Jede Zelle (diesmal tatsächlich ZELLE!) besteht aus zwei Polygonen, deren Hypotenusen immer von links

oben nach rechts unten ausgerichtet sind. Daher die verzerrte hexagonale Pyramide nach dem Anheben des Schnittpunktes. Probiert jetzt einfach mal herum, und kombiniert die erlernten Methoden untereinander. Übung macht den Meister!

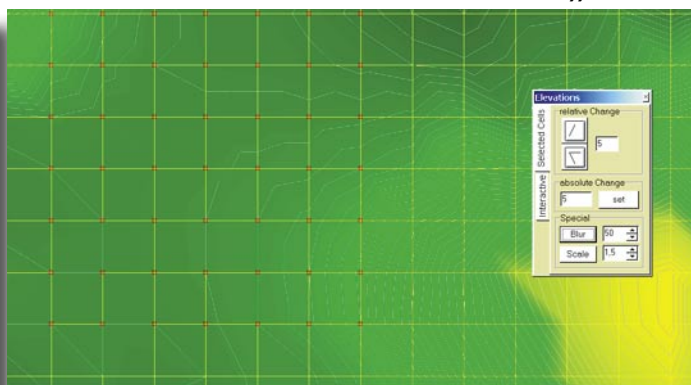
Neu in WT ist nun, dass man seine Insel mit „**Blure**“ zu besserer Performance verhelfen kann, bzw. allzu kantiges Gelände glätten kann. Wann immer die Map allzu uneben ist, benötigt OFP später deutlich mehr Rechenleistung zur Darstellung. Das lässt sich mit dem Blure-Button beheben. Das Eingabefeld daneben legt fest, wie stark die Funktion auf das Gelände wirken soll. Das funktioniert übrigens entweder für die ganze Map (alles markieren), oder nur auf selektierte Bereiche. Das Glätten lässt sich beliebig oft hintereinander durchführen – aber Vorsicht: ein „Rückgängig“ gibt es – wie bei allem anderen auch – nicht! Speichert daher vorher die Map, damit eine allzu starke Glättung wieder rückgängig gemacht werden kann. Neu ist auch die „**Scale**“-Funktion, mit deren Hilfe man seine Insel in der Höhe „strecken“ oder „stauchen“ kann. Das Eingabefeld legt den Faktor fest: Werte >1 strecken die Insel, oder selektierte Bereiche, Werte <1 stauchen die ganze Insel, oder selektierte Bereiche. Doch seid behutsam mit Werten >1 , denn ehe man sich's versieht, beginnt die Insel zu „clippen“, wenn die maximale Höhe (wir erinnern uns: 1474m maximal) überschritten werden sollte. Zudem wird die Insel – oder Teile davon – wieder „buckliger“, und OFP könnte wieder ins ruckeln kommen. Es macht also Sinn, zuerst die Insel zu „scalen“, und DANACH die Blure-Funktion anzuwenden.



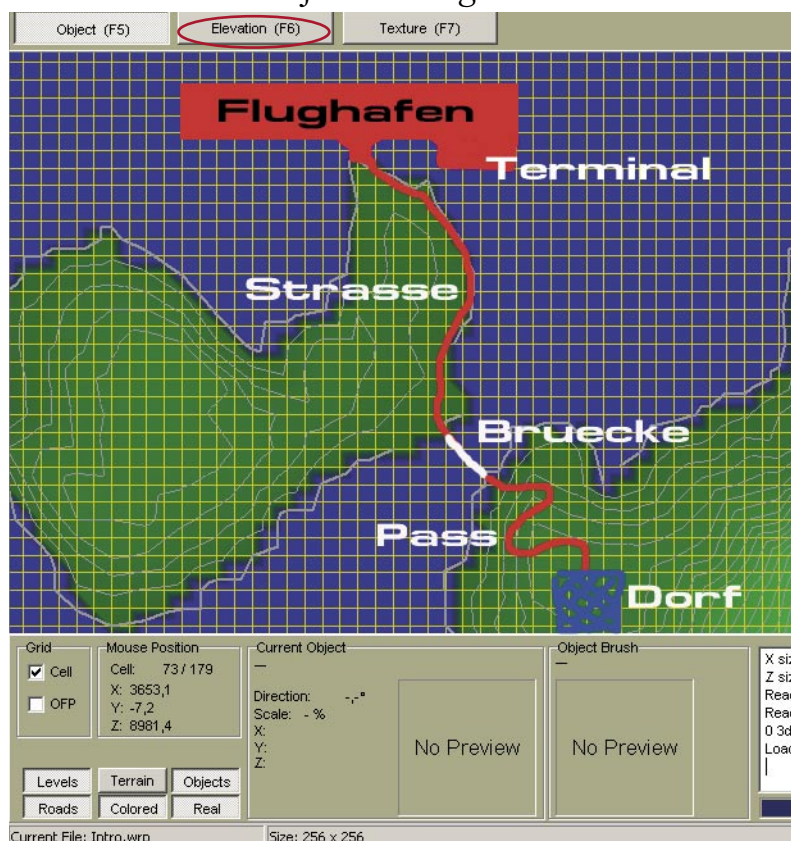
Vor „Blure“



Nach „Blure“



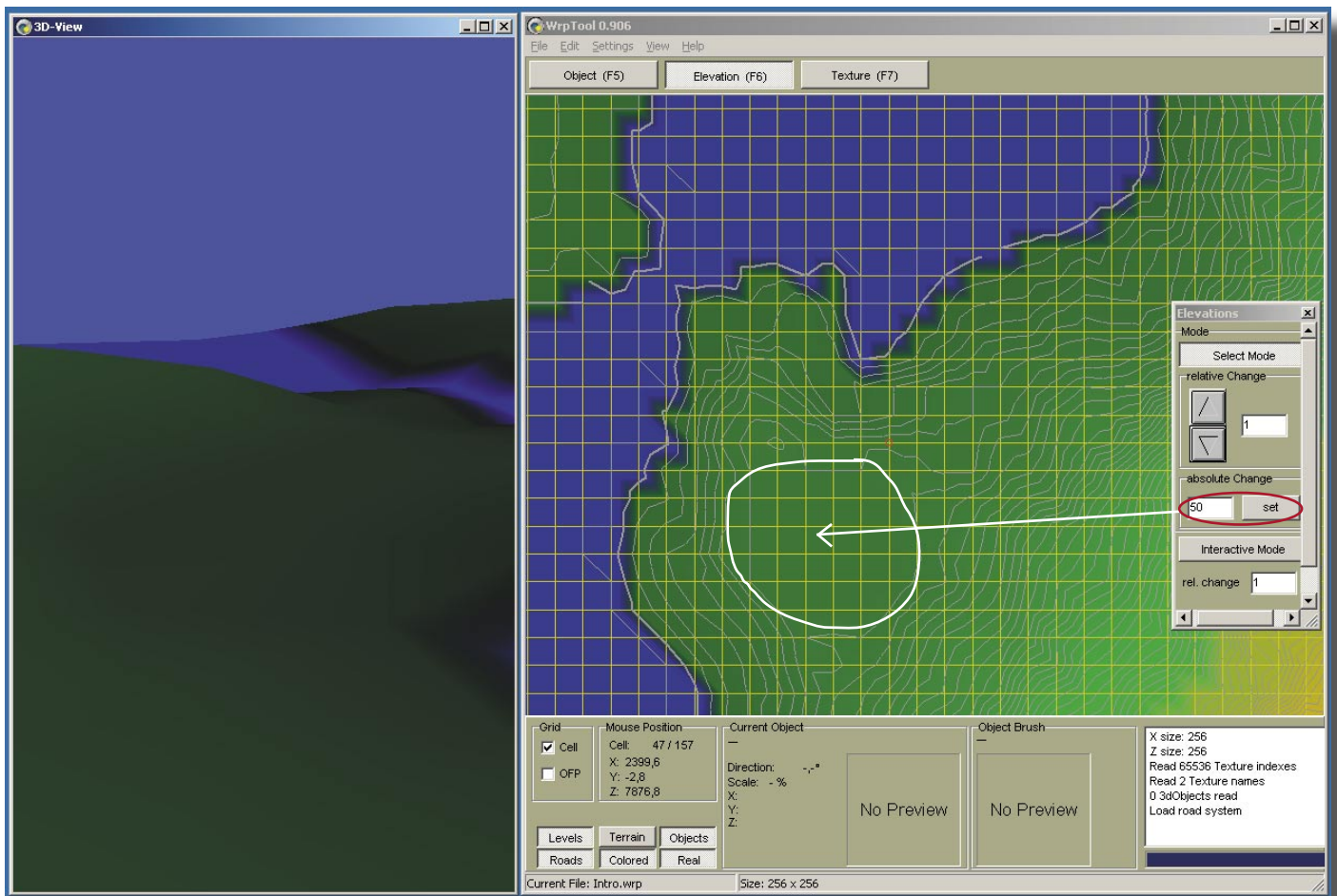
Bemühen wir jetzt wieder unsere Übungs-Map von vorhin, und versuchen jetzt unsere neuen Fertigkeiten auf die Aufgabenstellung anzuwenden. Wir wollten unsere Beispielin-sel für eine spätere **Infrastruktur** (Straße, Brücke, Flughafen, Dorfplatz) vorbereiten. Sucht also auf eurer Insel einen geeigneten Platz, und falls ihr dort noch keine Nachbarinsel (zwecks Brückenbau) haben solltet, dann „baut“ jetzt eine in WT – ihr wisst ja jetzt, wie das geht. Fertig soweit? Gut. Das Problem der geringen **Wassertiefe** (dürfte bei den meisten auftreten) wird auf ganz einfache Weise behoben: Alle Zellen der Map markieren, bei „**Relative Change**“ ca. 5 eingeben, und dann EINMAL auf den „**Pfeil-nach-unten-Button**“ geklickt – schon haben wir 7,2 Meter Meerestiefe, und das sollte reichen. Um nun gezielt arbeiten zu können, brauchen wir noch ein paar Infos, die wir uns aber leicht selbst errechnen können. Es geht um die **Zellengröße** der Map. **Wichtig:** die Zellengröße der Map ist nicht identisch mit der Zellengröße von OFP! Aufpassen also, und immer prüfen, welche Grids gerade aktiv sind! Wir wissen, dass z.B. eine Standard-Insel 256^2 Zellen, und eine Kantenlänge von $\sim 12,6$ Km hat. Teilen wir also die Kantenlänge durch die Anzahl der Zellen, erhalten wir die Kantenlänge einer Zelle in Km – Richtig? Macht also nach Adam-Riese aufgerundete 0,05 Km, oder 50 m. Für die diagonale Distanz multiplizieren wir noch mit $\sqrt{2}$, und erhalten dann ~ 70 Meter von „Schräg nach Links“. Ferner ist es gut zu wissen, dass die etwas dickere Linie, die eine Insel umgibt, auf Meereshöhe liegt. Natürlich ist das nur ein Mittelwert, denn Ebbe und Flut schwanken ja um ein gewisses Maß um diese Höhe. Nachdem nun ein **Brückenelement** aus Resistance ebenfalls 50 Meter Länge (welch glückliche Fügung) hat, können wir jederzeit genau abschätzen, wie wir unsere Insel im Detail manipulieren müssen, um unsere Vorstellungen umsetzen zu können. Mit diesen Infos ausgestattet, machen wir uns also ans Werk. Rechts seht Ihr, wie mein Übungsplan für die Beispiemap aussieht.



Anhand der Zellen können wir z.B. erkennen, dass wir voraussichtlich ca. 5 Brückenelemente benötigen werden. 3 Zellen auf Meereshöhe und diagonal macht $3 \times 70\text{m} = 210\text{m}$, zuzüglich einen Überstand, um auf Trassenhöhe das Land zu erreichen. Macht also wenigstens 5 ($=250\text{m}$) Brückenelemente notwendig. So verhindert man, dass man sich (und das passiert schnell mal) gewaltig in der Planung verschätzt. Das gleiche gilt für den Flughafen: Eine Runway-Kachel hat $50 \times 50\text{m}$, der Taxiway unmittelbar daneben ebenfalls. Macht schonmal 100m Breite, und damit's nach was aussieht, links und rechts mindestens noch einmal eine Kachelbreite ebenes Gelände. Gehen wir davon aus, dass eine Runway mit $1,3\text{ Km}$ Länge gewiss nicht überdimensioniert ist, sind dann „Summasummarum“ wenigstens 4 Zellen Breite und ~ 26 Zellen Länge notwendig, die ich für einen ordentlichen Flughafen einplanen muss. „**Runway-Kachel**“ bedeutet übrigens, dass OFP für Flughäfen ebendiese Runways (und auch Taxiways) als **Texturkacheln** bereitstellt. Es gibt sie für OW-WO-Richtung, und NS-SN-Richtung. Wer davon abweichen möchte/muss, kann diese Kacheln NICHT verwenden – Texturen lassen sich nicht drehen. In diesem Fall muss auf Objekte zurückgegriffen werden – entweder normale Strassen, oder – sofern vorhanden – Runway-Objekte. Die lassen sich dann in jede gewünschte Richtung drehen.

Ok, schreiten wir zur Tat, und beginnen mit dem **Flughafen**. Das Gelände dafür existiert bisher ja noch garnicht, also erschaffen wir eines, und zwar – wie im Plan angedeutet – im Meer. Nachdem wir uns vergewissert haben, dass wir uns im **Elevation-Mode (F6)** befinden, markieren wir – wie vorhin berechnet – 4 Zellen in N-S-Richtung, und ca. 25 Zellen in W-O-Richtung. Als nächstes heben wir das markierte Gelände über den Meeresspiegel, indem wir im Elevation-Tool bei „**absolute Change**“ eine 8 eingeben, und dann auf „**Set**“ klicken. Nun noch die Verbindung zum Land, sowie das Terminal-Areal markiert, und auf die gleiche Höhe gebracht. Voilà, unser Flughafengelände ist fertig vorbereitet. Machen wir uns also an die Anbindung zur Zivilisation. Dazu schlage ich erstmal vor, auf derselben Höhe, wie der Flughafen zu bleiben, und einfach der Küstenlinie zu folgen. Dazu ist ein wenig Fittzelarbeit nötig, denn wir müssen eine Menge Zellen einzeln markieren, damit das klappt. Wie das aussehen sollte, ist rechts zu sehen.





Um die Feinarbeit an der **Brücke** kümmern wir uns erst, wenn wir diese dann tatsächlich bauen. Momentan ist nur wichtig, dass die beiden Trassen-Enden einigermaßen aufeinander zielen, und etwa gleich hoch sind. Zunächst nochmal was leichtes, bevor es richtig knifflig wird – unser **Dorfplatz**. Wir benutzen wieder die Maus als Höhenmessgerät, und lassen sie über den zukünftigen Dorfplatz wandern. In meinem Fall zeigte WT an, dass das Gelände auf ca. 50–60 Meter Höhe liegt. Ok, wir gehen vor wie beim Flugplatz, nur diesmal stellen wir 50 Meter für die markierten Zellen ein. Somit müssen wir also für die Straße vom Dorf zur Brücke etwa 40 Meter Talwärts überwinden. Im Einzelnen heißt das, dass wir den Hang mit den Pfeil-Auf/Ab-Tasten in WT Meter für Meter, und Zelle für Zelle so bearbeiten müssen, dass ein passables Gefälle, aber mit möglichst wenig Seitenneigung, entsteht. Je nach persönlichen Ansprüchen gelingt dies mehr, oder weniger flott.

Ich möchte meinen, dass wir nun genug darüber erfahren haben, wie außer in Wilbur auch in WT das Terrain manipuliert werden kann. Zeit also, Entscheidungen zu treffen: Passt das Inselterrain (damit sind nicht solche Anpassungsarbeiten wie in der Übung vorhin gemeint) mehr oder weniger noch nicht, muss dem JETZT abgeholfen werden, vor allem dann, wenn man lieber nochmal zu Terragen oder Wilbur zurückkehren möchte.

Texture-Mode

Der Texture-Mode hat nun eine geniale Neuerung erfahren. Die Funktion „**Create Transitions**“ im Menü „**Edit**“. Das Texturieren unserer Insel wurde dadurch um vieles einfacher und vor allem schneller gestaltet. Auf die Einzelheiten gehe ich weiter unten noch ein. Schalten wir WT also mit „**F7**“ in den **Textur-Modus** (oder Button oben). Wie Textur-Modus und Textur-Browser funktionieren, wissen wir ja schon (Siehe **Vorbereitungen**). Außer ein paar Kleinigkeiten gibts dem nicht mehr viel hinzuzufügen.

Das Handling im Texture-Mode ist recht einfach: „**Shift**+**LMB**“ verteilt eine ausgewählte Textur auf alle markierten Zellen. Wenn ich eine Textur in eine oder mehrere Zellen einfügen möchte, geht das mit „**Einfg**“. Solange diese Taste gedrückt wird, wird die ausgewählte Textur in jede Zelle eingefügt, über die ich meine Maus bewege! Praktisch, nicht wahr? Alternativ kann ich auch mit „**Alt**+**LMB**“ eine Textur setzen. Wenn ich zu einer Textur wechseln möchte, die ich schon einmal auf der Map verwendet hatte, brauche ich nicht im Textur-Browser danach zu suchen (kann später mühselig sein), sondern bewege meine Maus einfach über die gewünschte Textur auf der Map, und drücke „**Pos1**“. Nun ist die Textur ausgewählt (auch im Texture-Browser), und ich kann diese nun woanders einsetzen. Macht euch ein wenig mit diesen vertraut.

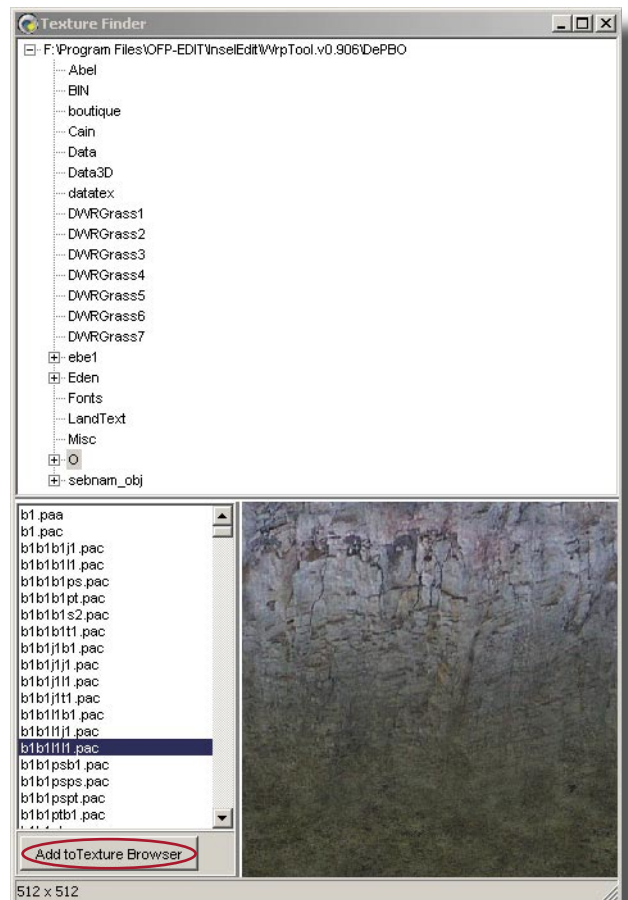
Texture-Browser



Der **Texture-Browser** (Menü „**View**“ oder „**Strg**+“**T**“) birgt ebenfalls keinerlei Geheimnisse. Wird eine schon texturierte Insel geladen, werden dort sämtliche verwendete Texturen angezeigt – was eine ganz ordentliche Menge sein kann. Startet man mit einer „jungfräulichen“ Insel, so ist – wie sicher schon bemerkt – nur die Textur für das Wasser vorhanden. Neue Texturen werden einfach mit dem Button „**Add New**“ (unten) hinzugefügt. Beim Klick auf den Button öffnet sich der...

Texture-Finder

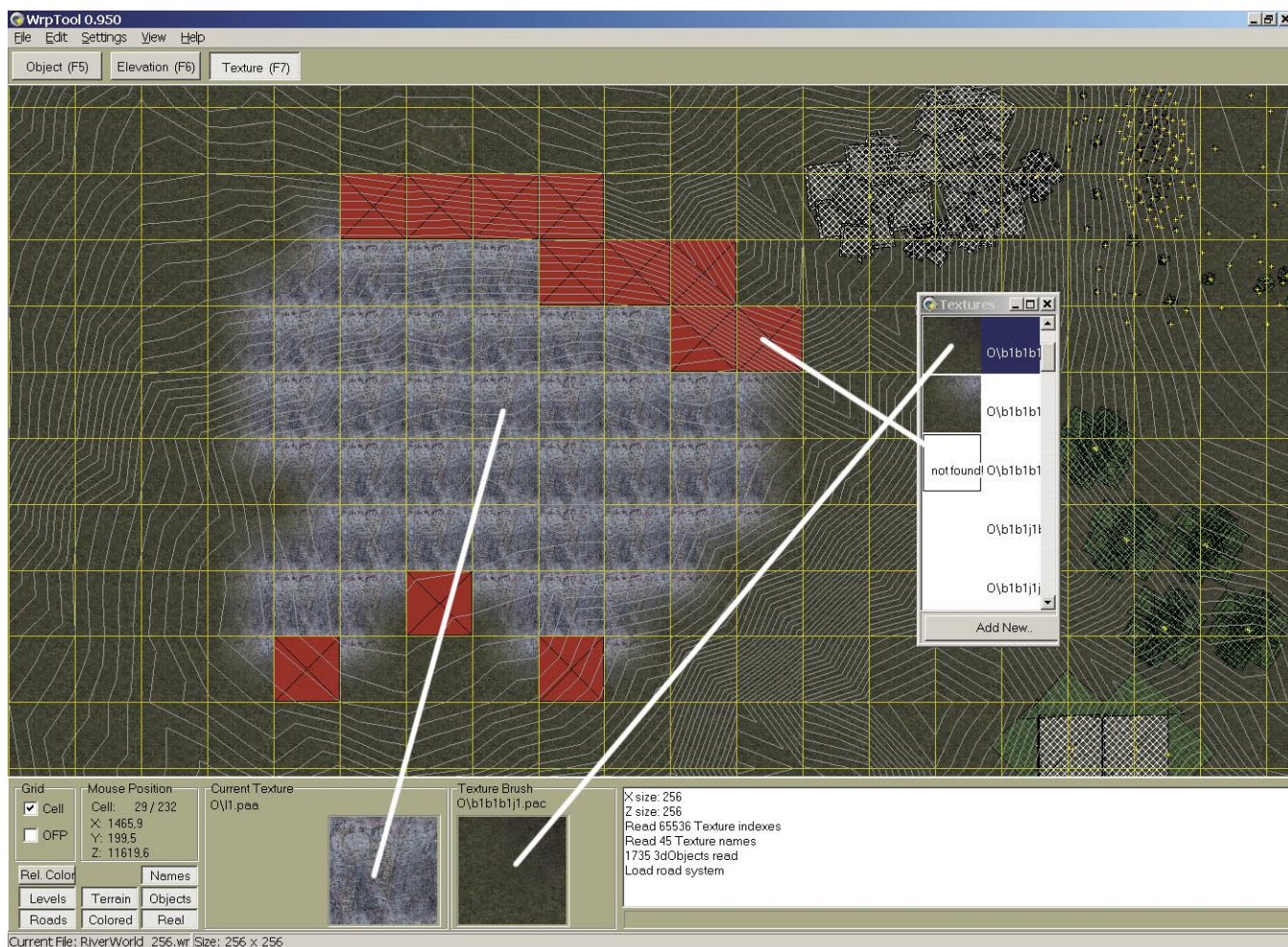
Auch der Texture-Finder ist ganz einfach zu bedienen. Normalerweise wird beim Öffnen schon der richtige Pfad oben angezeigt. Ein Klick auf das „+“-Kästchen öffnet den Inhalt. Nicht in jeder Kategorie sind auch Texturen enthalten, aber sobald welche vom Finder entdeckt werden, werden diese links unten aufgelistet. Wählt man dort nun eine Textur an, erscheint rechts daneben eine Vorschau. Sollte diese zu klein erscheinen, so lassen sich die Fenstergrößen durch verschieben der Trennbalken verändern, und so die Texturanzeige vergrößern. Das ist insofern praktisch, weil es einige sehr ähnliche Texturen gibt, deren Unterschiede sich im kleinen Browser-Fenster oft nicht erkennen lassen. Hier hat man jedoch die Möglichkeit, eine Textur sehr groß zu „zoomen“, um etwaige Zweifel an der Zugehörigkeit zu beseitigen. Mit gedrückter „**Strg**“-Taste kann man mehrere beliebige Texturen auswählen, und per „**Add to Texture Browser**“ diesem hinzufügen.



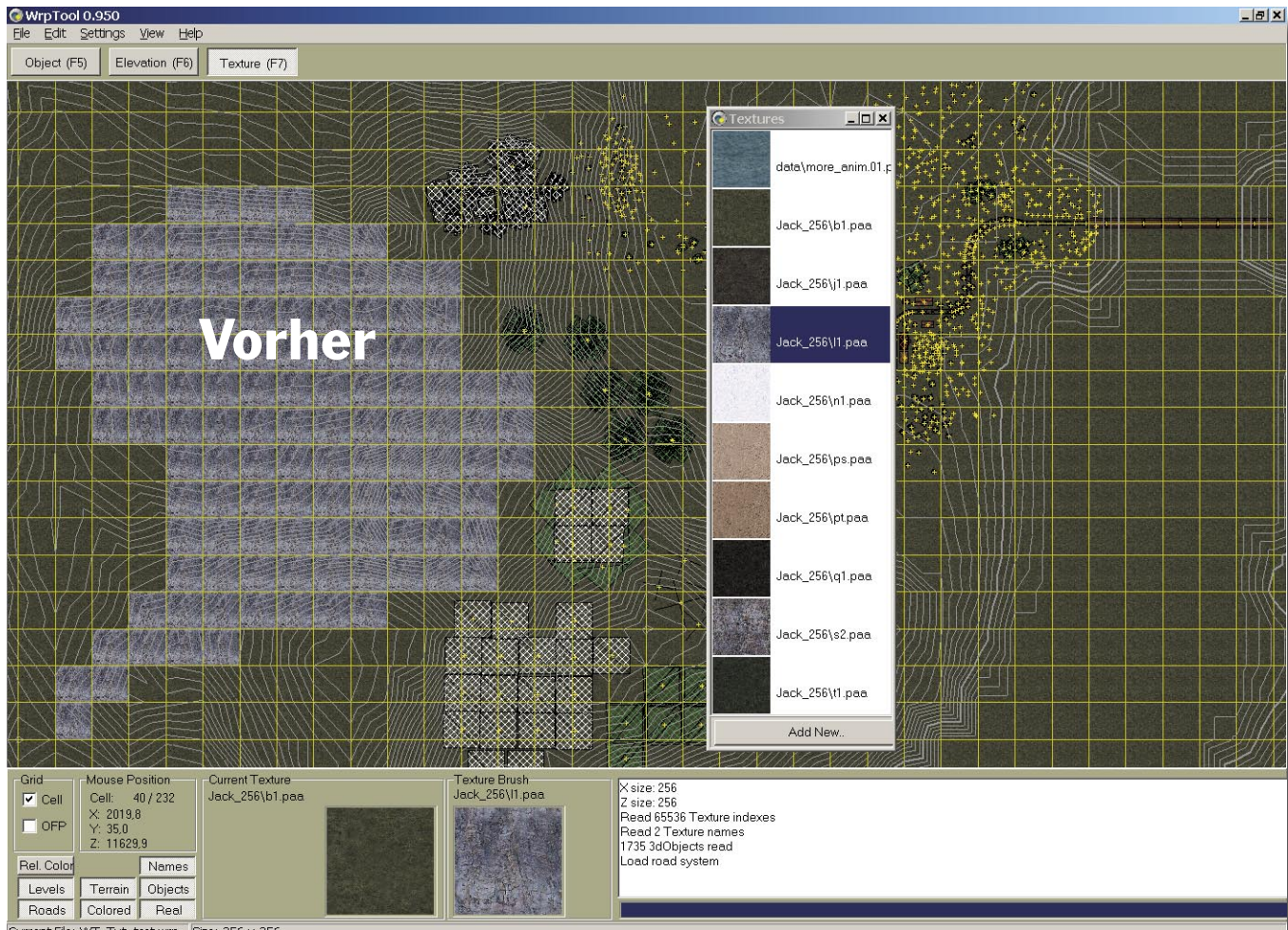
Wissenswert ist auf jeden Fall, wie das **Texture-System** von OFP funktioniert. Zentrales Element ist immer die Haupttextur. Diese hat maximal zwei Zeichen als Namen: z.B. „**b1.paa**“; „**s3.paa**“; „**js.paa**“; usw. Die Übergangstexturen hingegen haben immer acht Zeichen im Namen: z.B. „**b1b1b1s3.pac**“; „**b1s3b1js.pac**“; usw. Die Übergangstexturen sind in vier Untersektionen unterteilt, und jedes Zeichenpaar entspricht einem Viertel einer Haupttextur. Wir brauchen also, wenn wir eine Haupttextur-Kachel in eine andere Textur setzen, mindestens acht verschiedene Übergangs-Texturen (könnte man die Texturen drehen, wäre manches einfacher, doch leider geht das nicht). Doch das ist noch nicht alles. Schließlich gibt es ja noch andere Textur-Typen, in die wir unsere (immer noch dieselbe!) Textur-Kachel setzen könnten. Auch dafür brauchen wir passende Übergangskacheln. Und um das Maß endgültig

vollzumachen, kommt es nicht so selten vor, dass sich an einer X-Beliebigen Koordinate bis zu vier verschiedene, und auch noch unterschiedliche (da ist ein Unterschied!), Texturen treffen können. Wenn wir wollten, könnten wir nun leicht errechnen, wieviele Übergangskacheln also vorhanden sein müssten. Tun wir aber nicht, sondern behalten dies nur im Gedächtnis, und **vermeiden(!)** in Zukunft möglichst, „alte“ Texturen (Abel, Cain, Eden) mit Resistance-Texturen („O“-Ordner) auf ein und der selben Insel zu mischen! Glaubt es mir, das Chaos wäre perfekt. Wer nicht 120%ig genau weiß, was, und warum er es tut, sollte sich an meinen Rat halten. Denn die Res-Texturen sehen nicht nur anders aus, sie haben auch eine andere **Auflösung**.

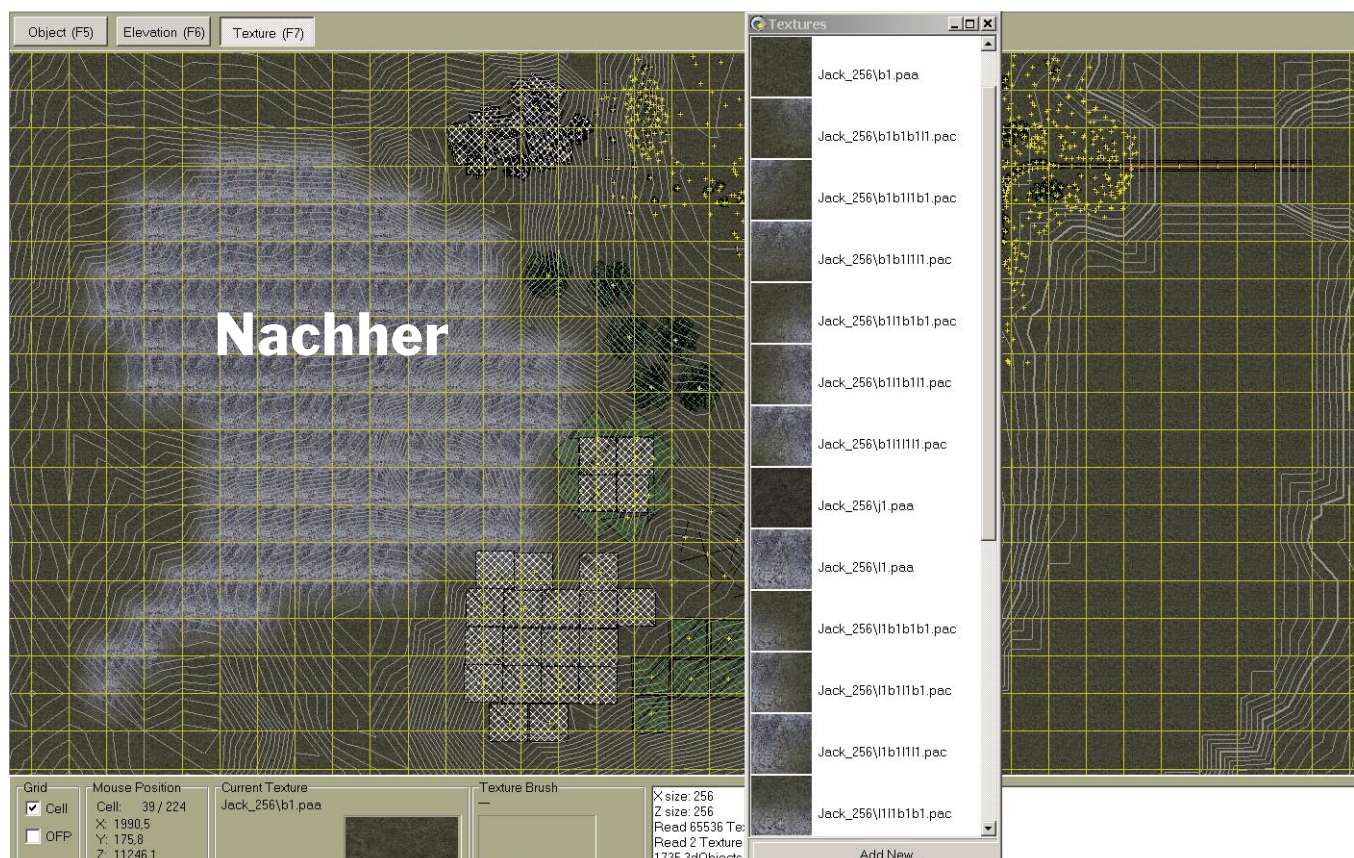
Hier gehe ich nun auf die neue Funktion in WT ein, sowie welche Vor- u. Nachteile damit verbunden sind. Naja, Nachteile hat die Funktion „**Create Transitions**“ (Transitions=Übergänge) eigentlich nicht - es ist eher ein Problem der Verfügbarkeit aller benötigten Texturen. Und die ist leider nicht gegeben! WT kann nur Übergangs-Texturen „berechnen“, wenn diese auch tatsächlich vorhanden sind - was leider nicht der Fall ist. OFP liefert ausschließlich jene Texturen mit, die auf den Inseln verwendet werden. Die rein theoretisch zusätzlich möglichen Texturen fehlen leider. Nun, das würde allein von der Dateigröße her auch kaum möglich sein - es würden hierfür mehrere HUNDERT!!! MEGABYTE zusätzlich benötigt. Um nun dieses Problem wenigstens ein kleines bisschen zu mildern, habe ich einen eigenen Textur-Satz erstellt, der die Fähigkeiten der neuen Funktion zumindest ansatzweise demonstriert. Dieser Textursatz liegt ebenfalls neben dem Tutorial zum DL bereit, und ich empfehle dringend, diesen auch zu verwenden - zumindest Anfangs, um das Üben etwas angenehmer zu gestalten. Ein paar Worte zum Textur-Set selbst: Es basiert auf den Resistance-Texturen, verwendet jedoch eine etwas geringere Auflösung. Optisch macht das kaum einen Unterschied, jedoch ist die Dateigröße um den Faktor 4(!) kleiner. Aus dem gleichen Grund existiert auch eine weitere Einschränkung: Es ist nicht möglich, mehr als zwei unterschiedliche Texturen an einem Ort zusammentreffen zu lassen. Sofern es die Zeit mir erlaubt, folgen später vielleicht noch weitere Sets, die diesen Mangel Stück für Stück beheben, doch momentan müssen wir uns mit den Einschränkungen zufrieden geben. Immerhin können wir nun jede beliebige Textur auf/neben einer anderen platzieren, und brauchen uns nicht mehr um das mühselige suchen der Übergänge zu kümmern. Wer allerdings partout mit den originalen Texturen operieren möchte, kann dies natürlich tun, und sich mit fehlenden Übergangstexturen herumschlagen.



Obiges Bild illustriert, was passiert, wenn man z.B. die Original-Resistance-Texturen verwendet, und WT per „Create Trasitions“ anweist, die Übergänge zu berechnen. Hier wurde die „l1.paa“-Grundtextur auf der „b1.paa“-Grundtextur platziert, und das Ergebnis war nicht sehr erfreulich. Da nützt es auch nichts, von Hand die Fehler ausbessern zu wollen, weil die benötigten Texturen schlicht und einfach NICHT VORHANDEN SIND! Doch betrachten wir die gleiche Situation mal mit dem neuen „**Jack_256**“-Set, das ihr hoffentlich nach beiliegender Anleitung schon installiert habt (wenn nicht, jetzt nachholen!). Eure Insel sollte ja schon mit einer Grundtextur versehen sein, aber höchstwahrscheinlich ist diese dem Original-Set entnommen. Das ändern wir jetzt einfach, indem nun aus dem „Jack_256“-Set alle **9** Grundtexturen in den Texture-Browser laden, und unsere Insel komplett markieren. Jetzt einfach eine Grundtextur auswählen, und mit gedrückter „Shift“-Taste auf die Map klicken. Nun sollte unsere ganze Insel mit einer „Jack_256“-Textur überzogen sein. Suchen wir uns nun einen Berg aus, und bedecken die Kuppe, sowie die steilen Hänge mit einer der beiden Felstexturen. Ihr braucht euch dabei nicht verkünsteln, es geht momentan ja nur mal ums ausprobieren, wie das ganze funzt. **WICHTIG:** WT kann nie mehr als **512** verschiedene Texturen verwalten! Mit „verschieden“ sind die Grundtexturen, als auch die Transitions gemeint, und davon kommen schnell etliche zusammen!

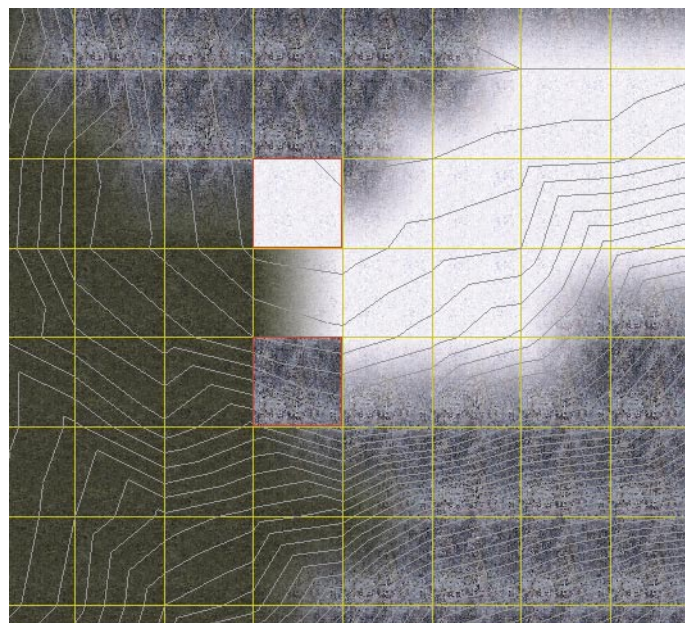


So in etwa sollte das ganze aussehen, bevor wir zum nächsten Schritt kommen. Im Menü „**Edit**“ klickt ihr jetzt auf „**Create Transitions**“, und schon röhrt WT los, um die passenden Übergangstexturen auszusuchen und zu platzieren – MEGACOOOL!



Doch nehmen wir mal an, es hätte geschneit, oder der Berg ist hoch genug für Dauerfrost. Dann möchte man doch gerne die Schnee-Textur verwenden – kein Problem, solange ihr euch dabei an die Restriktion der „2-Texturen-Regel“ haltet.

Wenn ihr das nicht tut, passiert das, was auf dem Bild rechts gezeigt wird. Wie es dazu kommt? Ganz einfach: WT kann keine eigenen Texturen erzeugen, sondern nur berechnen, welche Textur an welcher Stelle nach den Regeln erforderlich wäre. In diesem Fall erkennt WT, dass an zwei Orten eine Kachel mit je DREI verschiedenen Texturen einzusetzen wäre – doch für genau diesen Fall habe ich ja keine Texturen erzeugt – sie



fehlen dem Set. Also setzt WT stattdessen irgendeine der zwei Grundtexturen ein. Gut für uns, so erkennen wir sofort, wo der Fehler liegt, und können ihn ganz leicht beheben: Setzen wir einfach rund um die Schnee-Textur die Fels-Textur, und lassen WT nochmal die Übergänge berechnen – schon passt wieder alles. Behalten wir also im Gedächtnis: Immer nur zwei verschiedene Texturen zusammentreffen lassen! Alternativ bliebe momentan nur, mit den Originaltexturen ein wenig zu experimentieren, welche zueinander passen, ohne dass dabei Fehler entstehen. Wer will, kann ja auch die Original-Inseln eingehend analysieren, um festzustellen, welche Möglichkeiten zur Verfügung stehen.

Hier noch eine kleine Liste, welche Grundtexturen von Resistance sich (fast) ohne Einschränkungen paaren lassen:

Grundtextur: b1 > j1; s2; t1; pt (fast alle)

Grundtextur: j1 > t1; s2 (fast alle)

Grundtextur: l1 > keine

Grundtextur: n1 > s2 (fast alle)

Grundtextur: ps > pt; s2 (fast alle)

Grundtextur: pt > ps; s2; t1; b1 (fast alle)

Grundtextur: q1 > keine

Grundtextur: s2 > b1; pt; t1; p2 (fast alle); j1 (f. a.); n1 (f. a.)

Grundtextur: t1 > b1; j1; pt; s2

Noch ein Schlusswort zum Thema: Versucht möglichst nicht, Texturen mit großem Kontrast zueinander zu platzieren - es sieht meistens nicht soooo toll aus. Besser immer über wenigstens eine Zwischentextur von Hell nach Dunkel und umgekehrt wechseln. Und verwendet auch die beigegefügtten Kacheln, die auf manchen Grundtexturen für Abwechslung sorgen können.



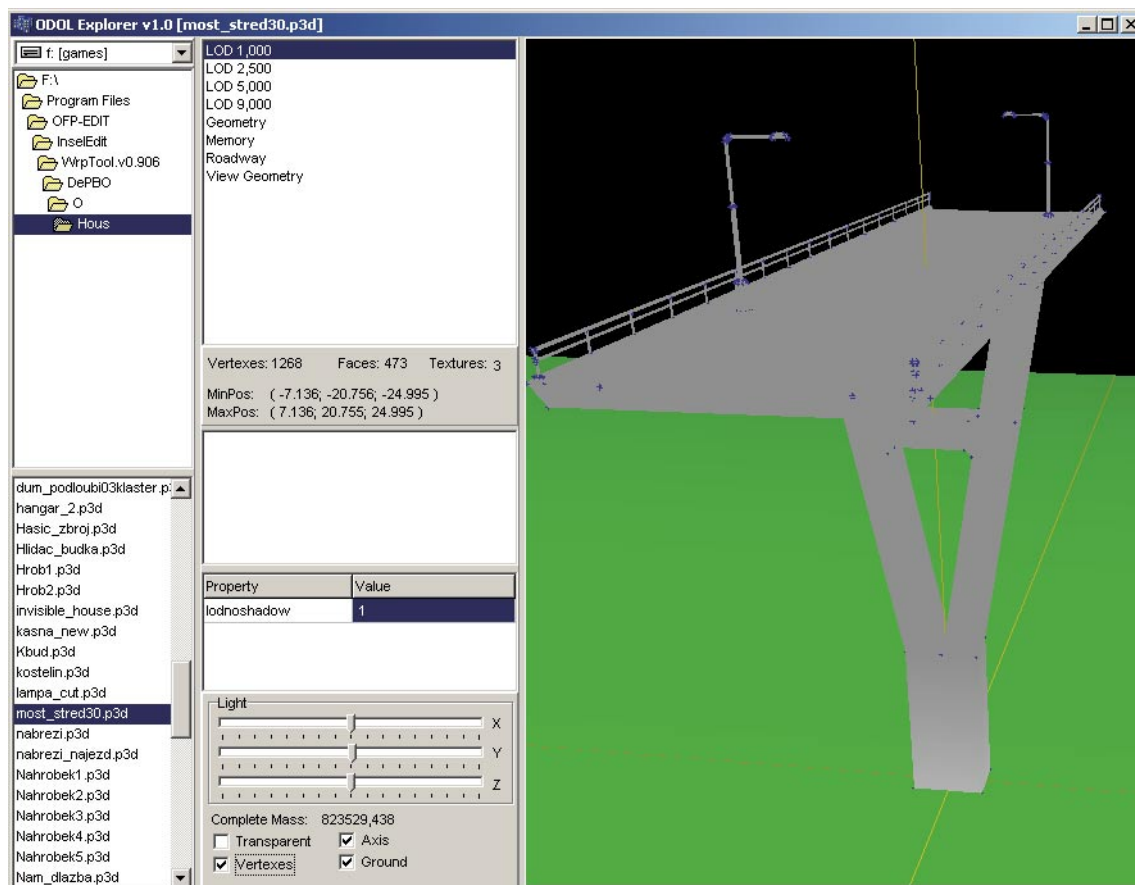
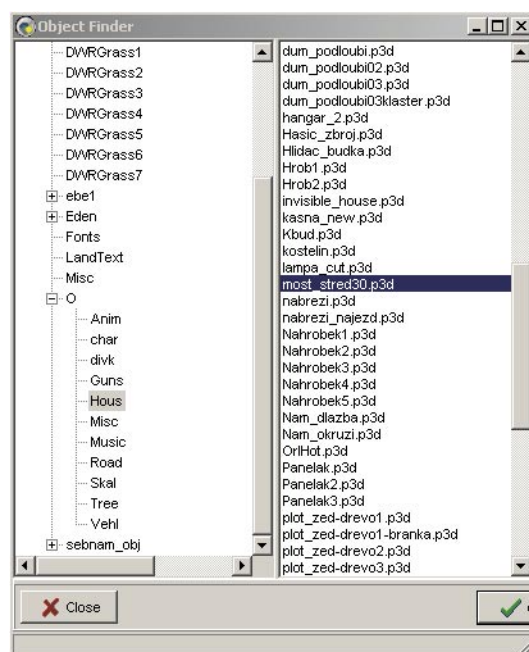
Objekt-Mode

Jetzt kommen wir endlich zu jenem Teil im Tut, den sicherlich die meisten sehnlichst erwartet hatten. Und das Beste am **Objekt-Mode** ist sicherlich, dass er der am leichtesten zu beherrschende Teil an WrpTool ist – wir können erstmal Aufatmen! :-) Schalten wir also mit „**F5**“, oder wieder mit dem Button oben, in den **Objekt-Modus**. Zoomen wir uns auf unserer Map auf ein beliebiges Stück Gelände heran, und beginnen damit, es zu „verschönern“. Öffnen wir als erstes mal den **Objekt-Browser** mit „**Strg**+“**O**“ (oder im Menü „**View**“), und betrachten das gute Teil. Eine ganze Portion der verschiedensten Kategorien sind erschienen, und ein Klick auf das „+“ öffnet eine Kategorie – das kennen wir ja schon. Die ominöse „**O**“ hinter den Einträgen sagt uns, dass davon noch nichts auf unserer Map platziert wurde. Setzen wir doch einfach mal 3 gleiche Häuser auf unsere Map (Haus anwählen, „**Alt**+**LMB** auf Map)....und schwupps, schon ist aus der „**O**“ eine „**3**“ beim entsprechenden Haus geworden. Diese Info ist für den einen nützlich, der andere schert sich weniger drum. Dem einen oder anderen mag aufgefallen sein, dass vor der jeweiligen Kategorie immer ein wenig Platz ist. Nun, das hat auch seinen **wichtigen** Grund: Wenn man in diesen Zwischenraum klickt, wird ein **Schloss** an dieser Stelle angezeigt. Jetzt kann das betreffende Objekt nicht mehr bewegt werden. Das ist insbesondere beim **Brückenbau** unumgänglich, doch dazu später noch mehr. Man kann auch eine ganze Gruppe sperren, indem man rechts neben das „+“-Zeichen einer Kategorie klickt (siehe rechts). Schlussendlich sehen wir noch ein Häkchen, sowie „**Random Dir.**“ daneben. Das bedeutet, dass bei gesetztem Häkchen jedes neue Objekt (auch wenn es sich immer wieder um dasselbe handelt) mit einer neuen Himmelsrichtung auf der Map platziert wird. Ergo entfernen wir das Häkchen immer dann, wenn wir z.B. eine Stadt bauen. Sollten dann noch kleine Unregelmäßigkeiten in der Ausrichtung gewünscht werden, macht man diese besser von Hand. Wie das geht, erfahren wir noch. Der „**Add New**“-Button unten öffnet den...

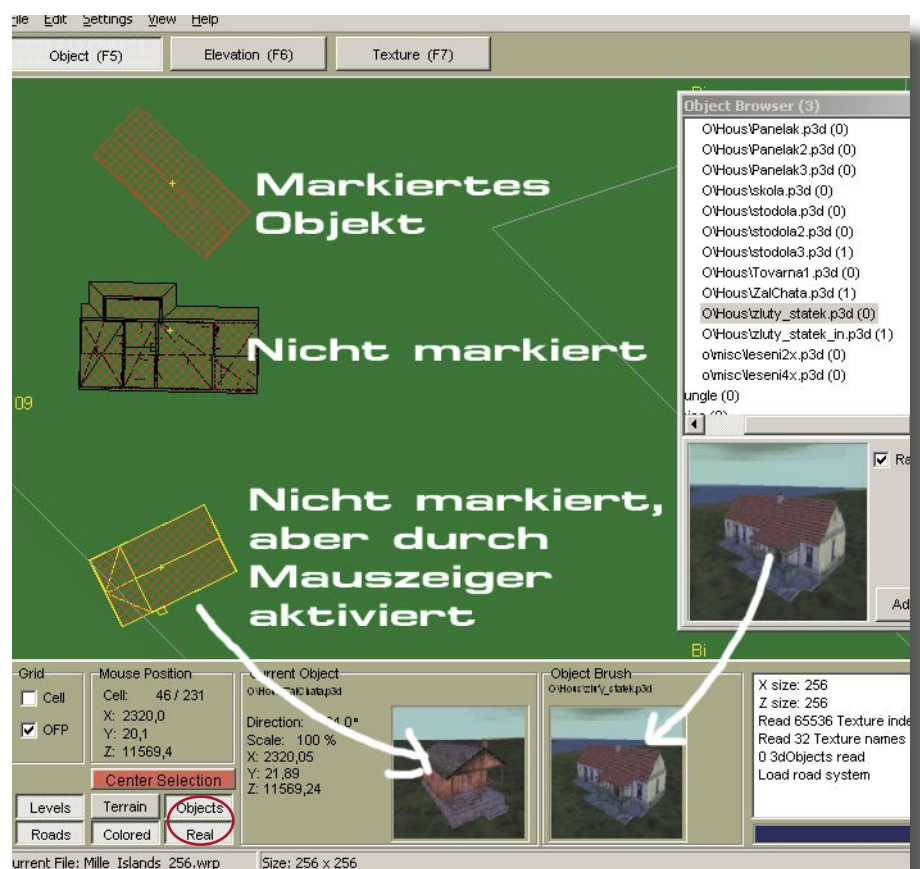


Objekt-Finder

Der **Objekt-Finder** ermöglicht es uns z.B. die Brückenelemente aus Resistance hinzuzufügen. Ferner ist der Objekt-Finder dafür zuständig, wenn wir Objekte aus inoffiziellen Addons laden wollen – genaueres hierzu erst im „**Update**“ des Tuts. Die Navigation im Objekt-Finder funktioniert genauso, wie bei den vorherigen Findern auch – mit einem Unterschied: Er hat keine Vorschau! :- (Doch gemacht, gemacht – dem wird abgeholfen. Es gibt ja noch den „**ODOL-Explorer**“, der dem Programm freundlicherweise beiliegt. Öffnen wir ihn doch mal (im Hauptordner von WrpTool), und voilà: Das ist unser Freund, denn mit ihm können wir jedes beliebige Objekt aus OFP betrachten! Suchen wir also z.B. die **Brücke** aus **Resistance**. Dazu stellen wir links oben den Pfad auf den „**O**“-Ordner in „**DePBO**“ ein. Wählen wir als nächstes den „**Hous**“-Ordner aus, und öffnen ihn. Nun müsste im unteren Fenster eine ganze Liste an Objekten erscheinen. Da wir ja



leider die genaue Bezeichnung der Brücke nicht kennen, müssen wir jedes einzelne Objekt in der Liste anklicken – solange, bis das Brückenelement im rechten Fenster erscheint. Sollte ein Ein- oder Auszoomen im 3D-Fenster notwendig sein, es funktioniert genauso wie beim 3D-Viewer („**Strg**“+**LMB**=Zoom, **LMB**=Drehen, **RMB**=Verschieben). Erscheint dann endlich das gesuchte Element im Fenster, merken wir uns die Bezeichnung, und suchen nach dieser nun wiederum im Objekt-Finder. Noch schnell auf „OK“ klicken, und „schon“ haben wir das Objekt im Browser zur Verfügung (ebenfalls wieder in der Kategorie „Hous“). Umständlich, nicht wahr? Aber dies ist die einzige **WIRKLICH** zuverlässige Methode, weil sie **IMMER** funktioniert. Klar, man könnte auch eine Map laden, auf der die gewünschten Elemente schon enthalten sind, dort danach suchen, Bezeichnung ablesen, merken, Insel wieder schließen, eigene Insel laden, Objekt-Browser öffnen, Add New klicken, Bezeichnung im Finder suchen, Ok klicken, und sich fragen, ob das jetzt wirklich schneller war?!? Und außerdem setzt diese Vorgehensweise immer voraus, dass ich genau weiß, wo ich nach was suchen muss. Der **ODOL-Explorer** macht also schon (zumindest jetzt noch) seinen Sinn. Es gibt schon Möglichkeiten, wie man sich vom ODOL-Explorer unabhängig machen kann, doch das „Wie“ wird erst im „Special-Teil“ des Tutorials erklärt – im Moment würde dies zu weit führen. Soweit so gut. Wir wissen also jetzt, wie wir noch nicht vorhandene Objekte zuladen können. Fangen wir also an, davon ein paar zu platzieren. Bitte seht jetzt nach, ob unten bei den 6 Ansichts-Buttons die beiden für „**Objects**“ und „**Real**“-Ansicht aktiviert sind. Die ersten 3 Häuser hatten wir vorhin ja schon platziert. Üben wir mit diesen also den Umgang mit Objekten auf der Map. Als erstes wollen wir mal eines der Häuser drehen. Dazu muss es markiert sein – einfach

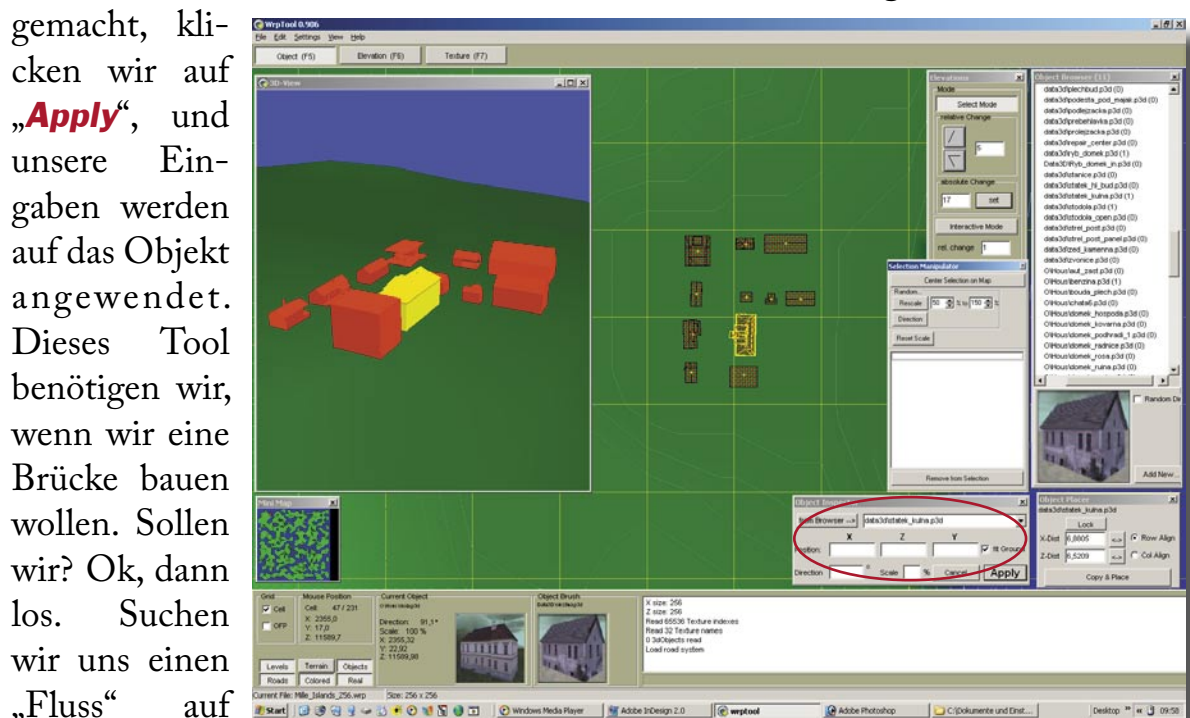


mit der gedrückten Maustaste über den Mittelpunkt ziehen, fertig. Das Haus-Symbol müsste nun komplett **Rot** sein. Seht euch das Beispiel oben genau an, und experimentiert ein wenig damit herum, damit ihr die jeweiligen Bedeutungen erkennt. Das linke Bild-Fenster unten zeigt euch immer das Objekt an, über dem sich gerade euer Mauszeiger befindet – in der Map leuchtet das Objekt dann gelb auf. Rechts daneben wird angezeigt, welches Objekt gerade im Objekt-Browser aktiv ist – dieses Objekt würde mit „**Ait**“+**LMB** auf der Map platziert werden. In der Mitte unten sehen wir alle wichtigen Angaben über das momentan auf der Map aktive – also **gelbe** – Objekt. Der „Center Selektion“-Button würde unsere Map über dem momentan markierten (das rote) Objekt zentrieren. Das ist dann von Nutzen, wenn ich mich kurzfristig von meinem gerade markierten Objekt entferne, und schnell wieder dorthin gelangen möchte. Im übrigen kann natürlich mehr als nur ein Objekt markiert sein. Dazu muss nur der Rahmen mit der Maus entsprechend größer gezogen werden. Mit „**Strg**“+**LMB** können diesem Rahmen noch weitere Objekte hinzugefügt, bzw. wieder abgewählt werden. Das Auswählen von Objekten sollte in Fleisch und Blut übergehen, es wird in allen Variationen sehr häufig gebraucht werden. Doch zurück zum Drehen von Objekten. Unser Haus ist also markiert, und wir halten nun die „**Shift**“-Taste gedrückt. Wenn wir jetzt irgendwo neben dem Objekt die Maustaste gedrückt halten, und ein wenig bewegen, dreht sich unser Haus schön brav ebenfalls auf der Map. Und weil's so schön war, markieren wir doch mal alle 3 Häuser, und versuchen das gleiche noch einmal – Hey, geht ja auch! Sehr schön, verschieben wir gleich mal unsere Häuser auf der Map. Dazu bewegen wir unsere Maus wieder so über das Objekt, bis dieses wieder gelb leuchtet. Jetzt können wir unser Haus – oder gleich mehrere, wenn markiert – mit **LMB** bewegen. Dabei fällt uns ein, dass wir eines der 3 Häuser wieder von der Map entfernen wollen – nix leichter als das: betreffendes Haus markieren, und „**Entf**“ drücken – wech isses. Richten wir unsere zwei verbliebenen Häuser nun ein wenig zueinander aus, und fügen noch ein paar weitere hinzu – sagen wir mal 10 - 12 Häuser, damit wir ein kleines Dorf bauen können. Wenn die Häuser stehen, fällt uns möglicherweise dabei auf, dass der Untergrund ein wenig zu steil abfällt? Kein Problem, wir haben ja schon gelernt, wie wir in WT den Boden planieren können. Kurz in den Elevation-Mode (**F6**) gewechselt, die Zellen markiert, die im Bereich des Dorfes liegen, und mit „**Absolute Change**“ auf gleiche Höhe gebracht. Das Beste daran ist, dass unsere bereits gesetzten Objekte sich ebenfalls automatisch heben, bzw. senken! Keine Korrekturen, kein Ärger, alles ist wunderbar.... außer,

man hätte vorher die Objekte „abgesperrt“, dann würden sie stur in der eingestellten Höhe verharren, und manchmal macht das ja auch Sinn. Man sollte halt nur immer daran denken, dass man im Objekt-Browser nachsieht, wenn etwas nicht so funktioniert, wie man will. Jetzt scheint auch die richtige Zeit, sich mit dem...

Objekt-Inspector

...zu befassen („**Strg**“ + „**I**“). Er ermöglicht uns eine exakte Manipulation der Objekte auf Eingabeebene. Wer z.B. bestimmte Objekte in exakten Positionen zueinander, bzw. auf der Map platzieren möchte, der kann dies in den beiden Koordinatenfenstern tun. Das 3. Fenster ist für die Höhe des Objekts zuständig, und damit sicher das Interessanteste für uns. Ebenso das „**Scale**“-Eingabefenster, das uns – und das ist eines der Highlights von WrpTool – das Skalieren von Objekten ermöglicht. Allerdings sollte man es dabei nicht übertreiben – die Entwickler geben eine Obergrenze von 290% an, und wir sollten es bei ~250% belassen. Andernfalls fällt solch ein „aufgeblasenes“ Objekt zu deutlich auf, und damit aus dem Rahmen. Kleiner geht bis runter auf 1%, aber Staub lässt sich sicher auch eleganter erzeugen! ;-) Halten wir also eine Untergrenze von ~25% als gerade noch Sinnvoll fest. Bei „Direction“ können wir das Objekt drehen, wobei „0°“ Norden entspricht, 90°=Osten, 180°=Süden, 270° oder auch -90° Westen, usw. Das Häkchen bei „**fit to Ground**“ sollte klar sein, und wir entfernen es, sobald wir davon abweichen wollen. Sind allen Eingaben gemacht, klicken wir auf „**Apply**“, und unsere Eingaben werden auf das Objekt angewendet. Dieses Tool benötigen wir, wenn wir eine Brücke bauen wollen. Sollen wir? Ok, dann los. Suchen wir uns einen „Fluss“ auf



der Map, über den wir unsere Brücke schlagen können. Solltet ihr in der Zwischenzeit eure Map geschlossen haben, fehlt das Brückenelement beim erneuten Laden der Map wieder. Also nochmal in den **Objekt-Finder**, und nochmal bei „**O/Hous**“ nach „**most_stred30.p3d**“ suchen und mit „**OK**“ „adden“. Jetzt dieses Element unter „**Misc**“ im Browser anwählen, und mit „**Alt**+**LMB**“ auf eine Flussseite platzieren – nur EINES, nicht mehr! – und richten es zum Fluss aus. Denn zum platzieren der restlichen Brückenelemente verwenden wir ein neues und sehr praktisches Tool, den...

Object-Placer

Unser Brückenelement muss markiert sein, damit der Objekt-Placer („**Strg**+**P**“) funktionieren kann (das gilt eigentlich fast immer, weswegen ich es in Zukunft nicht mehr erwähnen werde). Nun müssen wir noch herausfinden, welcher der beiden Radio-Buttons aktiv sein muss:

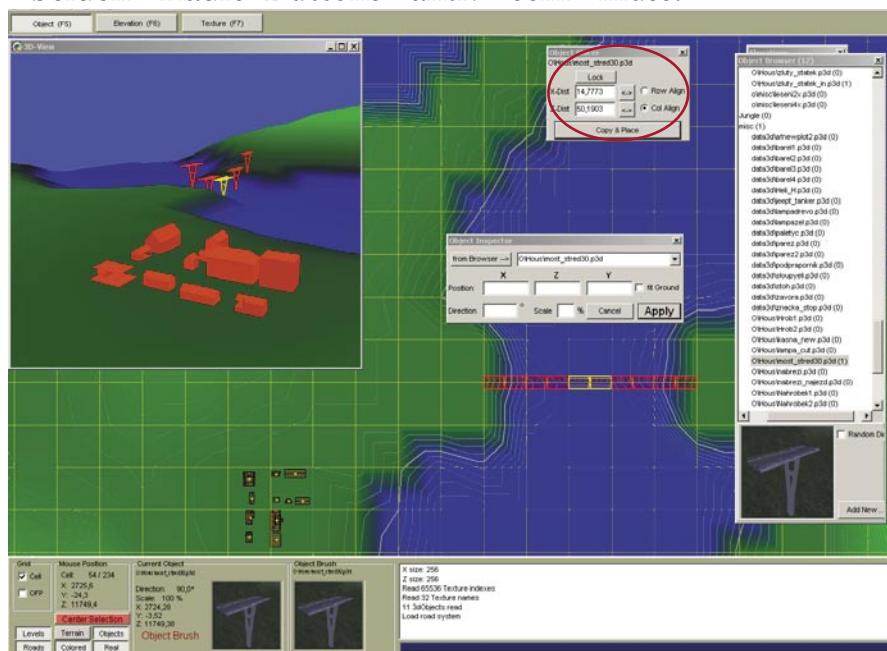
Cols = Zeilen,
Rows = Spalten

– ergo müssen wir „Cols“ wählen.

Wenn wir nun auf „**Copy and Place**“ klicken,

wird ein weiteres Brückenteil exakt an das vorherige „angedockt“. Klicken wir also so oft, bis der Fluss überwunden ist – praktisch, nicht

wahr? Kein Ausrichten, kein Fummeln...zumindest bis jetzt nicht ;–) Denn ein bisschen müssen wir schon noch selbst machen. Auf diese komfortable Weise ziehen wir z.B. unsere Zäune um Gärten oder Kasernen!

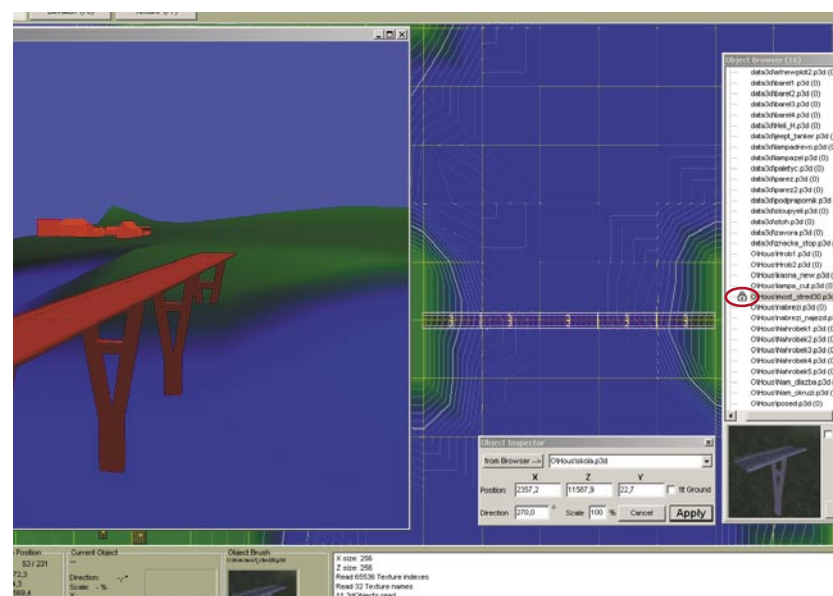
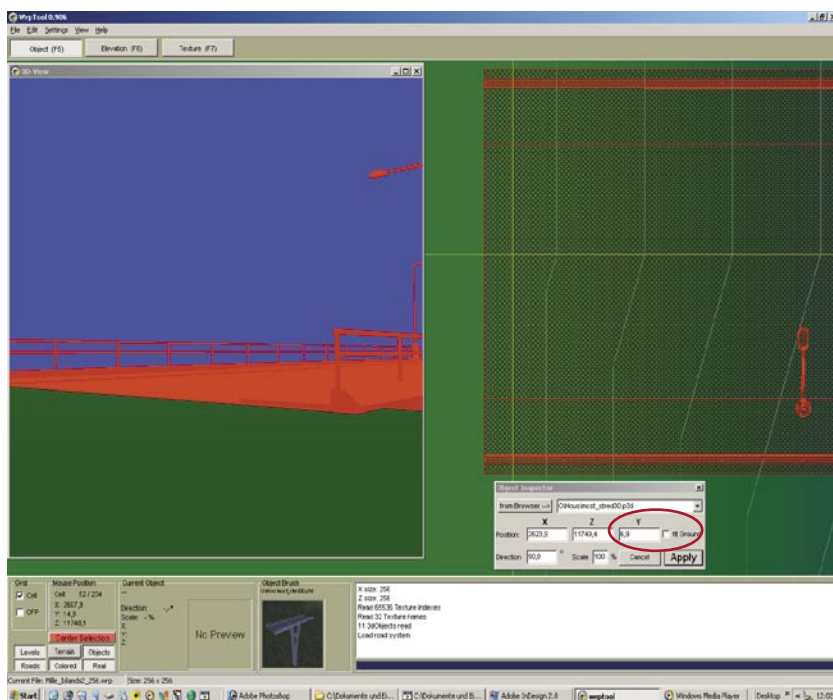


Wie ihr im 3D-Fenster sehen könnt, ist unsere Brücke bisher allenfalls für FreeClimber geeignet, nicht aber für so ein sensibles Gefährt wie unseren „Scoda“. Gehen wir also wieder zum **Objekt-Inspector**, und passen unsere Brücke nun

an das Ufer an. Es versteht sich von selbst, dass beide Ufer gleich hoch sein müssen. Gegebenfalls müsst ihr das im Elevation-Mode noch anpassen. Ebenso WICHTIG: Euer Meeresgrund sollte nicht mehr als ~5 Meter tief sein, und das Ufer nicht höher als ~20 Meter. Denn die Brückenteile MÜSSEN den Grund berühren – andernfalls funktioniert eure Brücke NICHT! Ein Brückenelement misst vom Grund des Pfeilers bis zur Trasse 33 Meter. Somit stecken in meinem Beispiel noch ca. 3 Meter von den Pfeilern im Meeresgrund – das genügt. Markieren wir also das erste Brückenelement, und entfernen das Häkchen bei „**fit to Ground**“ – jetzt ist das Eingabefeld nicht mehr grau hinterlegt, und wir können selbst bestimmen, was Sache ist. Probiert jetzt also einfach mal ein paar Werte durch, und aktualisiert mit „**Apply**“.

Alles klar? Gut. Für die Feinarbeit müsst ihr im 3D-Fenster ordentlich nah heranzoomen. Das Brückenelement (Strassenbelag) muss möglichst genau auf Terrainhöhe gebracht werden! Andernfalls entsteht ein „Achsen-Killer“ im Übergang zur Straße. Lasst euch nicht von den unterschiedlichen Darstellungen

bei verschiedenen Zoomstufen irritieren (je nach Zoomstufe passt die Höhe mal, mal wieder nicht). Das letzte bisschen checken wir erst nach der Kontrolle in OFP. Denn nur dort wird endgültig entschieden, was passt, und was nicht. Ok, wir haben unser erstes Brückenteil an das Ufer angepasst, gleichen wir nun auch die anderen an. Dazu lesen wir den Y-Wert aus, den wir beim ersten Teil verwendet haben, und merken uns diesen. Nun markie-

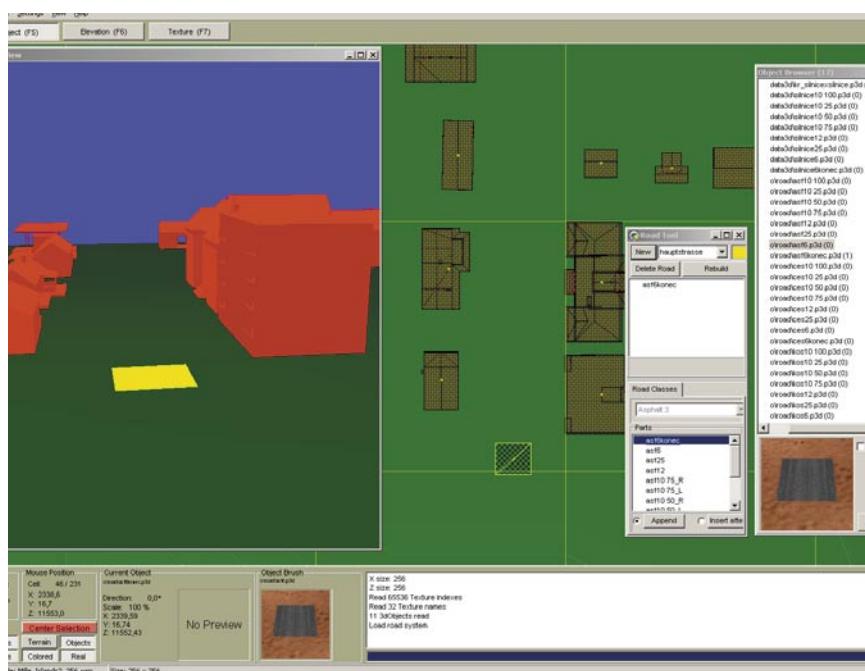


ren wir das nächste Element, und tragen dort im Y-Fenster ebendiesen Wert ein. Nach klick auf „**Apply**“ schwuppt das Brückenteil augenblicklich auf die gleiche Höhe wie das Erste. Diese Prozedur wiederholen wir also mit allen anderen Elementen, und wenn wir fertig damit sind, dann **SCHLIEßEN WIR DIE BRÜCKE** im Objektbrowser **AB!!!** Andernfalls – und das verspreche ich euch – war die ganze Arbeit für die Katz! Denn spätestens, wenn wir unsere erste Straße gebaut haben (was wir gleich tun werden), müssen wir vor dem Speichern nochmal alle Objekte „auf Grund“ setzen, da sonst die Straßen nicht richtig funktionieren. Hätten wir dann unsere Brücke(n) nicht gesperrt, würden diese ebenfalls in ihren ursprünglichen, und damit unbrauchbaren Zustand zurückversetzt. Also immer ans **Absperren** aller Objekte denken, die nicht auf dem Grund aufliegen sollen – ob höher, oder tiefer gesetzt, ist dabei egal.

Soweit, so gut – bauen wir also eine Straße vom Dorf zur Brücke. Gehen wir also wieder ins „**View**“-Menü und öffnen dort das...

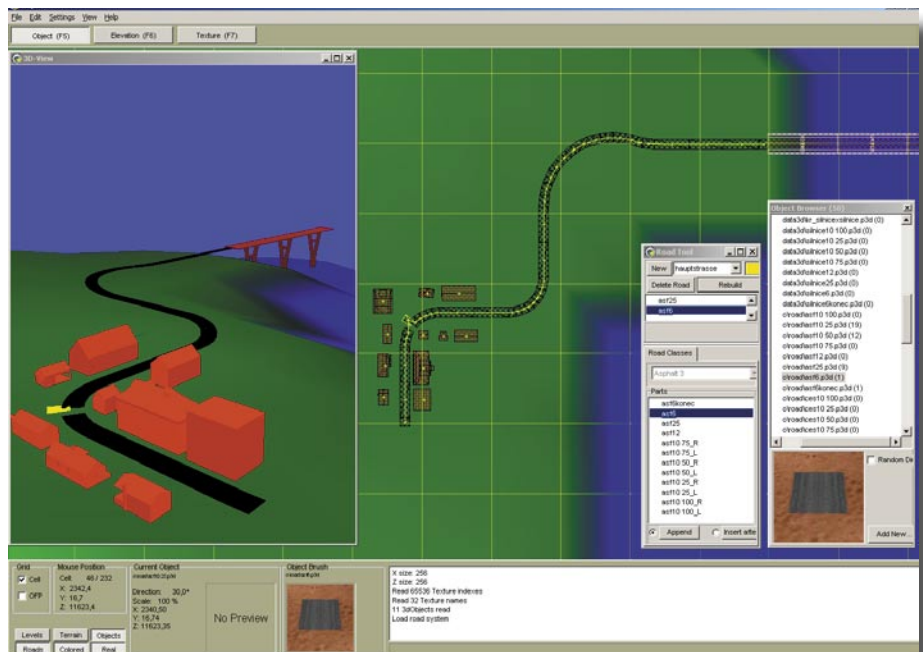
Road-Tool

Zuerst müssen wir im Objekt-Browser ein allererstes Stück Straße auf der Map platzieren. In der Regel beginnen wir mit einem sogenannten „Terminator“ – das sind die Straßenstücke, deren Textur auf einer Seite durchsichtig wird, um schließlich im „Nirvana“ zu verschwinden. Solche Teile haben immer den Zusatz „conec“ in der Bezeichnung. Wenn man aber eine in sich geschlossene Straße erstellen möchte, kann man natürlich auch mit einem normalen Straßenobjekt beginnen. Wer wissen möchte, welche Straße sich hinter den Bezeichnungen verbirgt, brauch nur im Objekt-Browser die Typen mit der „6“ in der Be-



zeichnung anklicken – schon gibts eine Voransicht. Das Roadsystem ist anhand der Bezeichnung übrigens ganz leicht auseinanderzuhalten. Gerade Stücke enthalten EINE Zahl, die der Länge in Metern entspricht. Kurven haben eine zweite Zahl dahinter, die zusätzlich über den Kurvenradius Auskunft erteilt. Das „R“, bzw. „L“ am Ende teilt uns die Richtung der Biegung in Baurichtung mit.

Sobald also unser erstes Stück Straße auf der Map „klebt“, markieren wir dieses. Sodann klicken wir im Road-Tool auf „**New**“, und geben dem Kind einen Namen: „Hauptstraße“ zum Beispiel. Nun hat Road-Tool uns in seinem unteren Fenster eine Liste (fast) aller Straßenstücke zur Verfügung gestellt, die wir benötigen. Zudem könne wir, wenn gewünscht, der Straße eine andere Farbe auf der Map verpassen, indem wir auf das gelbe kleine Feld rechts oben klicken. Wer viele Straßen gebaut haben wird, wird das zu schätzen wissen. Wenn wir jetzt mit der rechten Maustaste auf einen

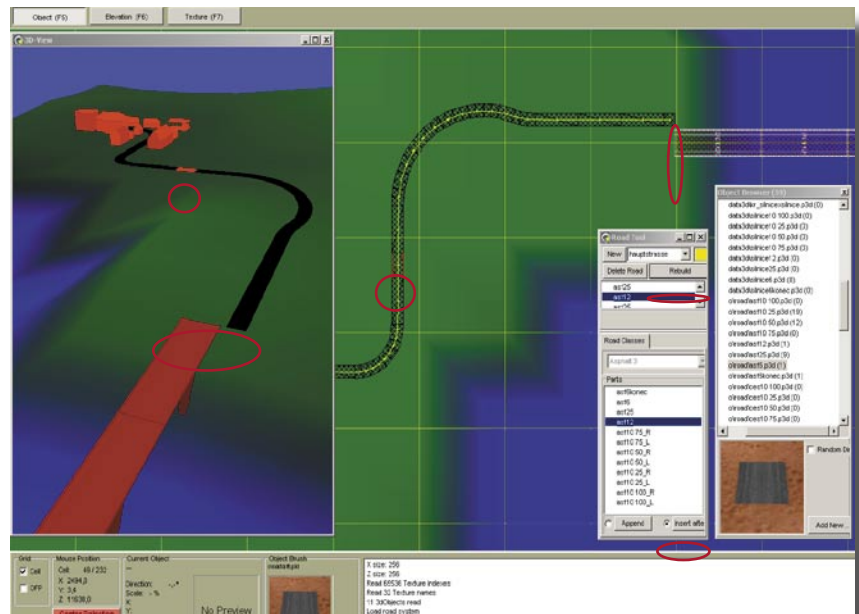


Straßentyp im unteren Fenster klicken, wird sofort in der Map das entsprechende Teil an das erste angefügt – und zwar exakt! :-) Ok, los gehz! Klick, klick, klick, und fertig ist die Straße! Lediglich das Anpassen an die Brücke erfordert wieder etwas Feingefühl. Normalerweise fängt man ja auch dort an, wo die genaueste Anpassung erforderlich ist, aber weil ich halt so ein fieser Charakter bin, hab ich euch mal wieder ein totes Pferd untergejubelt! ;-) Nein, im Ernst: das sollte nur der Übung dienen, und nicht, euch zu ärgern! Bevor also jemand die Nerven verliert, weil die Trasse „ums Verrecken“ nicht an die Brücke passen will, markieren wir einfach die ganze Straße mitsamt den Häusern in einem Rutsch, und schieben nun den ganzen Tross schön bequem an die Brücke ran. Ente gut, alles gut ;-)

Doch Hoppla! Hat sich doch während der Verschiebeaktion das eine oder andere Straßenstück dazu entschlossen,

so mir nix, dir nix aus der Reihe zu tanzen?! Dann bloß nicht dran rumfummeln, das geht „Ruckizucki“ mit einem Mausklick auf **„Rebuild“**! Und schon ist unsere Straße wieder wie geschniegelt.

Sollte jemand seine Straße fast, oder ganz fertiggestellt haben, und hernach irgendwann feststellen, dass irgendwo Teile hinzugefügt werden müssten, dann geht auch dies. Dazu schalten wir das Road-Tool auf „Einfügen“ um – das ist der Radio-Button rechts unten und heißt **„Insert after“**. Jetzt können wir das Straßenteil markieren, das VOR (immer in ursprünglicher Baurichtung gesehen) dem einzufügenden Stück liegt, und wieder wie gehabt weitere Stücke einfügen. Lasst euch nicht davon beirren, dass die Straße dabei immer mehr deformiert wird, denn wenn alle Teile eingefügt sind, klicken wir wieder auf **„Rebuild“**, und unsere Straße sieht wieder propper aus. Natürlich liegt nun das Ende der Straße ganz woanders – eben um genau dieses eingefügte Straßenteil entfernt vom ursprünglichen Ort.



Der eine oder andere mag sich mittlerweile gefragt haben, wo denn nun die Kreuzungen geblieben sind? Keine Bange, die sind alle da. Und zwar findet ihr diese im **Objekt-Browser**, wenn ihr nach Bezeichnungen mit **„kr_New“** sucht. Begegnen sich z.B. „silnice...p3d“ und „asfaltka...p3d“ auf eurer Map, so müsst ihr nach **„kr_new_sil_asf_t.p3d“** suchen, und diese dann (vorerst) von Hand einfügen. Vorerst deshalb, weil es mit dem Road-Tool noch mehr auf sich hat. Denn standardmäßig sind nicht alle Roads definiert, und auch die jetzt schon verfügbaren sind nicht komplett konfiguriert. Dies wird mit dem **„Road-Definer“** bewerkstelligt. Doch weil dies eine recht knifflige Angelegenheit ist, und wir ihn nicht sofort brauchen, habe ich ihn in eine neue Version des Tuts „verbannt“. Bevor wir mit irgendetwas anderem beginnen, sollten wir gleich jetzt folgendes machen: Öffnet das Menü **„Edit“**, und klickt auf den Eintrag **„Align all Obj. to Ground...“**. In dem folgenden Hinweis wird nochmal gefragt, und überdies darauf hingewiesen, dass abgeschlossene Elemente davon unberührt bleiben – genau das wollen wir ja auch. Also bestätigen wir

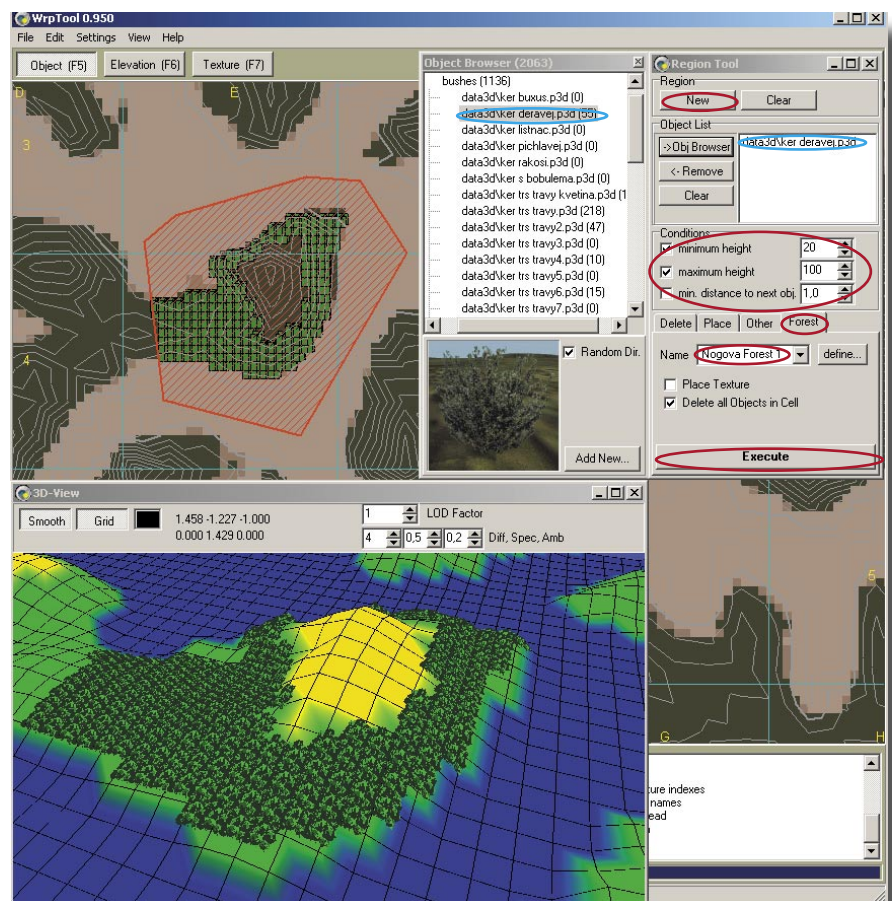
eingestellt? Gemach, gemacht – dieses Feature macht durchaus Sinn. Werft mal einen Blick auf den Screen oben – dort habe ich mehrere Objekte gleichzeitig markiert. Wenn ihr jetzt „**Strg**+„**C**“ drückt, und eure Maus an einen anderen Ort auf der Map bewegt, könnt ihr dann per „**Strg**+„**V**“ diese ganze Gruppe dort einfügen. Und jetzt kommts: Denn diese ist ja (noch) identisch mit der Quell-Gruppe, und das wollen wir nicht! Ergo brauchen wir nur noch auf „**Rescale**“ und auf „**Direction**“ klicken, und unsere Gruppe ist nicht mehr wieder zu erkennen! Nur die jeweiligen Typen sind noch dieselben, aber eben größer, oder kleiner, und anders ausgerichtet. Sollte man doch einmal die ursprüngliche Größe der Objekte wiederherstellen wollen, genügt ein Klick auf „**Reset Scale**“ – schon ist wieder alles auf 100% zurückgesetzt. Lediglich die Ausrichtung lässt sich nicht per Knopfdruck wiederherstellen. Dazu braucht's wieder den „Inspector“, oder aber Ausrichtung von Hand. Neu hinzugekommen ist, dass man nun bei „**Direction**“ selbst den Grad der Verdrehung bestimmen darf.



Wälder

Mannomann, was lag mir dieses Thema im Magen, es euch zu erklären. Wenn es ein Dauerthema in den einschlägigen Foren gab, dann das Problem mit den Wäldern! Doch was soll ich sagen - das Problem hat sich Dank der fleißigen Arbeit der Entwickler um einiges erleichtert :-)

Im Gegensatz zur vorherigen Version muss jetzt ein Wald nicht mehr einzeln zusammengebastelt werden, sondern kann mit dem „**Region Tool**“ fast vollautomatisch generiert werden. Jede Operation wird mit „**New**“ gestartet, das leitet das Markieren eines Bereichs ein, und wird durch einfaches Klicken mehrerer Punkte auf der Map erreicht. Der letzte Klick muss ein **Doppelklick** sein, dadurch wird die Linie zu einem schraffierten Polygon geschlossen. Um das Besondere des neuen Tools möglichst schnell zu verdeutlichen, wählen wir einen Bereich, der sowohl Meeresgrund, wie auch größere Höhenlagen umschließt. Natürlich wollen wir weder auf dem Meeresgrund, noch auf den kahlen Höhen eines Berges Bäumchen pflanzen. Damit ebendies nicht geschieht, teilen wir WT eben unsere Wünsche mit, indem wir bei „**minimum height**“ ~20m, und bei „**maximum height**“ ca. 100m eintragen. Doch noch hat WT nicht alle benötigten Parameter, es muss noch wissen, welchen Wald es platzieren soll. Klicken wir also auf den Reiter „**Forest**“ ein Stück darunter, und wählen einen der fünf Wälder aus. Der vorletzte Punkt verlangt eine kleine Zwischenüberlegung: Bevor wir wissen, ob wir das Häkchen bei „**Place Texture**“ setzen sollen, oder nicht,



müssen wir verstehen, was es bewirkt, und welche Probleme es mit sich bringen könnte. „Place Texture“ würde einfach die jeweilige „Wald-Textur“ unter den Wald legen. Bei einem Everon-Wald wäre es also auch die Everon-Wald-Textur aus dem CWC-Textur-Set. Alles gut und schön soweit – doch was, wenn wir unsere Insel mit Res-Texturen belegt hätten? Dann könnten wir hernach keinesfalls mit „Create Transitions“ die richtigen Übergänge erzeugen, weil es schlichtweg dafür keine gibt! Ergo müssten wir in diesem Fall auch das Häckchen bei „Place Texture“ entfernen, und die passende Res-Wald-Textur von Hand unter den Wald legen. Hätten wir jedoch eine Novoga-Wald gewählt, könnten wir sowohl bei Res-Texturen, wie auch beim „Jack“-Set das Häckchen setzen. Unterm Strich muss halt immer beides zusammenpassen, damit das Häckchen gesetzt sein darf. Haben wir unsere Entscheidung getroffen, können wir auch schon auf „Execute“ klicken, und – voilà – haben wir einen prächtigen Wald, der weder im Wasser, noch auf felsigen Gipfeln wächst. Hinterher brauchen – wenn nicht schon geschehen – nur noch die Wald-Texturen gesetzt werden, und per „Create Transitions“ angeglichen werden. Nach diesem Schritt fällt sicher auf, dass WT die Texturen scheinbar um eine Raster nach rechts und nach unten verschiebt, und das ist tatsächlich so. Ist aber nicht schlimm, wir markieren einfach unseren neuen Wald, und ziehen ihn einfach ebenfalls ein Stückchen nach rechts unten. Sollte dieser Weg versperrt sein, erweitern wir von Hand unsere Textur nach links und oben und lassen nochmal berechnen.

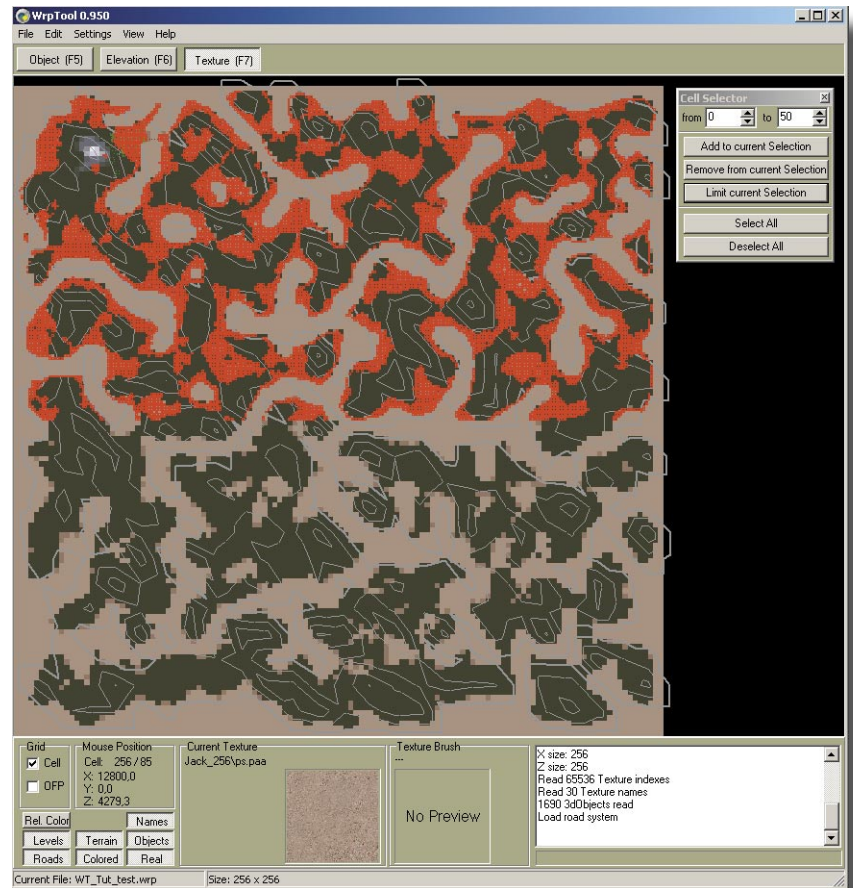
Natürlich war das nicht alles, das Tool kann mehr. So wie mit dem Wald geht das auch mit allen anderen Objekten – wer will, könnte nun den vielzitierten Schilderwald mit ein paar Mausklicks entstehen lassen! Aber ich denke, Büsche und Gräser werden bei den meisten mehr Gefallen finden ;-) Ferner lassen sich auch ganz neue Waldtypen neu definieren – ganz ähnlich wie bei den Straßen. Aber auch hier schiebe ich das Thema erstmal ins Abseits, um den Rahmen nicht zu sprengen. Außerdem will ich ja den Forschernaturen unter euch nicht jeglichen Spaß nehmen! ;-)

Cell Selector

Noch ein neues und nützliches kleines Tool für WT. Es funktioniert beim **Elevation-** u. **Textur-Modus** – für den Objekt-Modus ist das Region-Tool zuständig. Warum ist dieses Tool

nun so nützlich? Ganz einfach: Wenn wir die Natur betrachten, fällt uns nicht selten auf, dass jede Höhenlage ihre ganz eigene Vegetation oder Bodenbeschaffenheit hat. Und genau das können wir nun auf einfache Weise für unsere Insel nachbilden. Beginnen wir mit dem Strand als Beispiel: Es soll vom Meeresboden bis hinauf auf ~ 20m Höhe eine Sandtextur platziert werden, und danach bis auf 100m Grasland, bis auf 300m Wald, und dann nur noch Fels befinden – nichts leichter als das! Doch zunächst ein

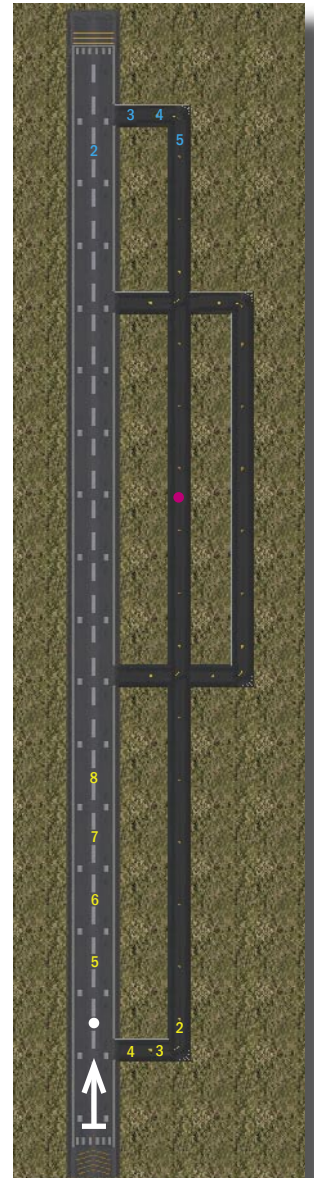
Wort zum Bild: Ich habe dort zur Veranschaulichung zuerst eine Hälfte der Map ganz normal markiert, bevor ich das Tool zum Einsatz brachte. Genausogut hätte ich dies auch lassen können. Das bedeutet, dass ich das Tool entweder auf die ganze Map, oder auch nur auf markierte Bereiche anwenden kann. Für unser Beispiel gelten die Operationen für die ganze Karte. Fangen wir also mit



dem untersten an, und geben „-50“ nach „**from**“, und „20“ vor „**to**“ in die Felder ein. Danach klicken wir auf „**Add to current Selection**“, und schon ist der gewünschte Bereich markiert. Nun können wir unsere Textur einfügen, und sofort weitermachen. Dazu geben wir nun „20“ ins erste Feld ein, und „100“ ins zweite. Ein Klick auf „**Limit current Selection**“ sollte den neuen Bereich erfassen. Falls nicht, einfach nochmal alles „**deselectieren**“, und wieder auf „Add to current Selection“ klicken. Das funzt auf jeden Fall. Nun wieder die Textur aufbringen, und dann wie gehabt weitermachen, bis alles texturiert ist. Natürlich sollte man dieses Grundmuster an einigen Stellen durch Aufbringen leichter Variationen auflockern, das macht die ganze Insel noch natürlicher. Ganz genauso kann man das Tool auch zur **Höhenmanipulation** benutzen – z.B. um bestimmte Bereiche zu „soften“. Also ein rundum nützliches Werkzeug, das mit einfacher Bedienung glänzt.

Flugplatz

Noch so ein Hammerthema, das ich nun versuche, wenigstens Ansatzweise rüberzubringen. Es mag dem einen oder anderen aufgefallen sein, dass zwar Runways für beide Richtungen existieren, aber nur die N/S- bzw. S/N-Version Taxiways zur Verfügung stellt. Das liegt daran, dass die KI nur in diesen beiden Richtungen landen und starten kann. Das Warum kenne ich nicht, aber so ist es halt. Ferner ist in diesem Beispiel genaugenommen durch die Texturgestaltung schon die Richtung der Starts und Landings festgelegt, nämlich von Süd nach Nord – umgekehrt geht zwar auch, wäre aber entgegen den Markierungen der Bahn. Vielleicht finden unsere Script-Füchse heraus, dass es auch in anderen Richtungen (W-O/O-W) klappen könnte, aber für unser Tut ist schon diese eine Richtung Herausforderung genug. Versuchen wir uns also erstmal klarzumachen, was die KI alles benötigt, um überhaupt vernünftig agieren zu können. In der **Config.cpp** werden vier Kategorien definiert: „**ilsPosition[]**“, „**ilsDirection[]**“, „**ilsTaxiIn[]**“ sowie „**ilsTaxiOff[]**“. Zunächst müssen mal der Start- u. Aufsetzpunkt bekannt sein. Den habe ich rechts mit einem weißen Punkt markiert – das ist der „ilsPosition[]“-Punkt. Doch mit diesem einen Punkt kann die KI noch nicht viel anfangen – sie benötigt auch die Richtung; sie wird später in der Config.cpp festgelegt. Der zweite wichtige Punkt ist der **Center-Point**, der definiert werden muss – ich habe ihn hier pink markiert. Um diese beiden (und auch die noch folgenden) Punkte für die Config.cpp verfügbar zu machen, brauchen wir nur die Maus auf die jeweiligen Punkte zu führen, und dann die Koordinaten links unten in WT herauslesen und notieren – alternativ können auch vorübergehend Objekte dort platziert werden, das Ergebnis ist das gleiche; aber dem einen oder anderen mag diese Methode einfach angenehmer sein. Zurück zu den noch fehlenden Marken. Unser AI-Flieger steht also noch immer am Center-Punkt, und fragt sich, wo's lang geht. Die gelben Ziffern für die Taxi-Off-Prozedur weisen im den Weg zum Startpunkt, und sogar noch ein Stückchen darüber hinaus. Das stellt sicher, dass er sich während dem Abheben auch schön brav über der Startbahn orientiert, und sich nicht schon nach dem ersten Lupfer granatenmäßig in die Kurve legt. Sollte das immer noch der Fall sein, könnt ihr die Punkte auch ein wenig weiter auseinanderziehen.



Ok, unser mutiger KI-Pilot kann nun also zur Startbahn rollen und zackig abheben. Doch holen wir ihn gleich wieder vom Himmel, wir wollen schließlich eine saubere Landung und hernach ein aufgeräumtes Flugfeld. Wo unser Flieger landen soll, weiß er schon, nämlich wieder auf unserem „ilsPosition[]“-Point. Das allein genügt aber wieder nicht, unser KI-Pilot würde noch recht planlos ausrollen, und dann auf der Runway stehenbleiben. Damit er sich zurechtfindet, orientiert er sich an den blauen Markern, die wir ihm zur Verfügung stellen. Nun ist euch sicher aufgefallen, dass sowohl die gelben, wie auch die blauen Marker erst mit „2“ beginnen. Das ist deswegen so, weil beim Start der **Center-point** die „1“, und beim Landen der „ilsPosition[]“-Point die „1“ ist. Zusätzlich sucht der KI-Pilot beim Landen nach der 5-er-Marke den Center-point, und kommt dort zum stehen – Landebahn aufgeräumt, alles in Butter :-)

Na gut, zugegeben, das war noch nicht alles, aber immerhin der größte Teil. Es gilt nur noch, die jetzt bekannten und ermittelten Punkte in die Config.cpp zu übertragen. Zu diesem Zweck kopiert ihr euch den ganzen ILS-Teil aus einer schon existierenden Insel heraus, und setzt ihn in eurer eigenen Config.cpp ein. Hernach werden die Werte einfach durch eure eigenen ersetzt – fertig ist die Laube. Das einzige, was ihr noch kontrollieren müsst, ist die schon erwähnte **Richtung**. Die wird im Teil „ilsDirection[]“ definiert, und zwar durch ein Vorzeichen bei der letzten Ziffer im Werteblock – eine „1“ bedeutet N-S-Richtung, eine „-1“ die **gegenteilige** Richtung. In unserem Fall müssten wir also eine „-1“ eintragen.

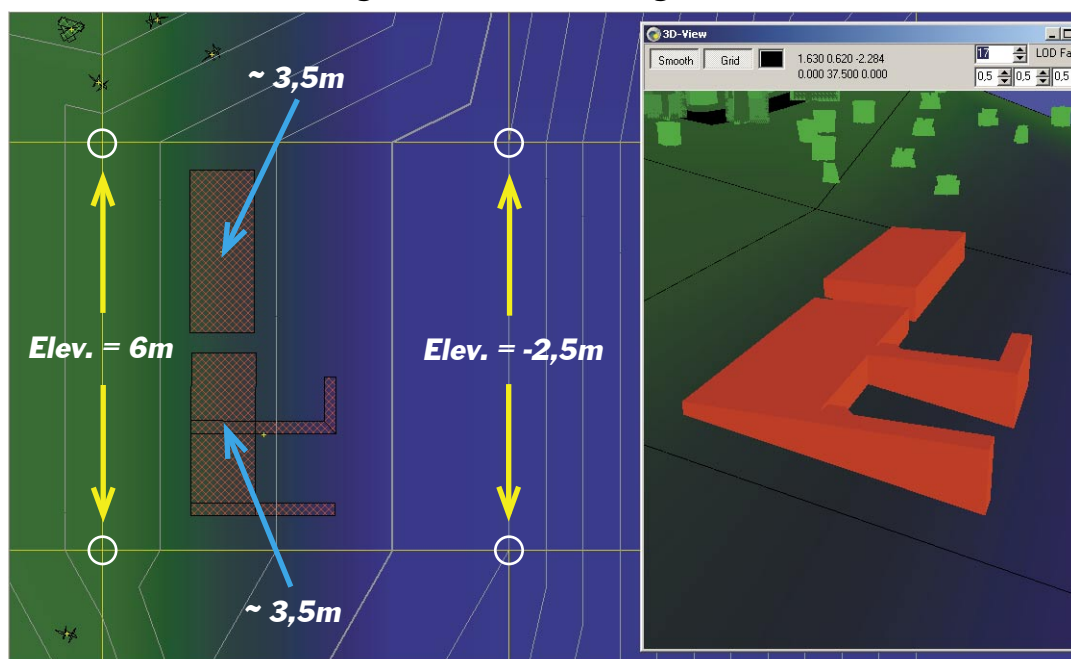
Seht zu dieser Angelegenheit bitte auch im Original-Manual nach, vor allem was die .cpp betrifft. Lasst euch nicht irritieren, der Autor dort hat für sein Beispiel nur die gegenteilige Start- u. Lande-Richtung gewählt – sonst ist alles genauso wie bei mir. Mir ist klar, dass meine Beschreibung den meisten zu dürftig sein wird, aber ich möchte dazu anmerken, dass ich selbst aus Zeitmangel noch keinerlei Erfahrungen mit Airports und deren .cpp sammeln konnte, sowie dass dieses Tut sich vorerst hauptsächlich an den Anfänger richtet. Und das erstellen eines KI-fähigen Airports ist sicher keine Anfängermaterie. Ich bitte also um Nachsicht, und verweise alle besonders Mutigen auf die einschlägigen Foren, wenn es um die Klärung der sicher auftauchenden Fragen geht. Ein Tipp noch: Wer sich hier selbst richtig reinknien will, kann auch einen fertigen Flughafen nehmen, und auf den Koordinaten der Config.cpp einfach Objekte pflanzen. Damit werden die KI-Punkte deutlich für eine Analyse sichtbar.

Seehafen

Das Thema mit den Häfen mag vielleicht nicht jedermann interessieren, aber wer einen Hafen bauen möchte, wird auf ein Problemchen stoßen: den **Tidenhub**. D.H. bei Ebbe trocken-net unser Kai entweder vor sich hin, oder säuft bei Flut fast ab. Was also tun? Nun, eine perfekte Lösung wird es nicht geben, zumindest nicht, solange mit den BIS-Elementen gearbeitet wird. Hier wären die Modeler gefragt, die uns besser angepasste Hafenelemente liefern könnten. Vielleicht gibt es ja auch schon welche, dann lasst es uns wissen. Doch gehen wir mal davon aus, nur die BIS-Teile verwenden zu können. Damit wir mit diesen zu befriedigenden Ergebnissen kommen können, müssen wir eine Methode finden, diese so zu platzieren, dass diese einerseits nicht ganz trocken fallen, und andererseits nicht überflutet werden. Halten wir zuerst mal fest, welches diese Elemente sind:

data3d\molo_beton.p3d,
data3d\molo_krychle.p3d,
o\hous\nabrezi.p3d and
o\hous\nabrezi_najezd.p3d

Als nächstes sollten wir festlegen, wo unser Harbour entstehen soll, denn dort müssen wir einige Vorbereitungen mit den Höhenparametern treffen. Wir stellen mit dem Elevation-Tool den Strand wie unten gezeigt auf die jeweiligen Höhen ein, und platzieren unsere Hafenelemente so, dass die Zentren ungefähr bei Höhe 3,5m zum liegen kommen – fertig.

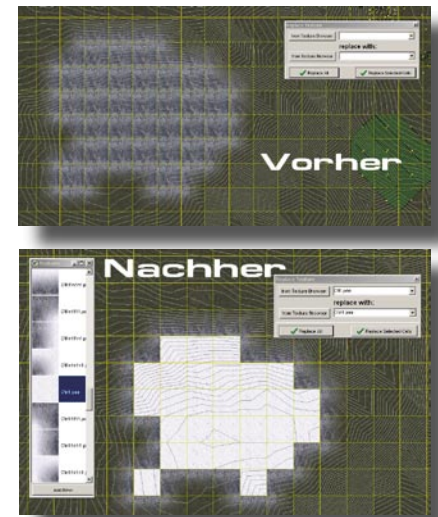


[illegible]

port“ klicken, und eine Box zum speichern öffnet sich. Vergeben wir also einen treffenden Namen mit einer „**.txt**“-Endung an, und speichern das Teil. Hernach können wir einfach das Textdokument öffnen, und seinen Inhalt in unsere **Config.cpp** kopieren – dorthin, wo sich die ganzen Namen sonst auch befinden (einfach in einer anderen .cpp nachspicken).

Obj. u. Tex. ersetzen

Angenommen, wir hätten unsere Insel fertig texturiert, aber wir wollen nun auch noch eine Winter-Version unserer Map. Damit diese aber nicht so langweilig aussieht, sollen die Fels-, und die Strandtexturen erhalten bleiben. Ok, dazu öffnen wir im Menü „**Edit**“ > „**Replace Textures**“, und wählen im **Texture-Browser** die Textur an, die wir ersetzen wollen. Jetzt können wir im Fenster „**Replace Textures**“ auf „**Add from Browser**“ klicken, und unsere Textur ist eingetragen. Gehen wir wieder in den Texture-Browser zurück, und wählen dort die neue Textur aus. Ein erneuter Klick auf „**Add from Browser**“ – diesmal der **untere** Button – trägt die neue Textur ein. Klicken wir nun auf „**Replace all**“, werden alle Felstexturen auf der Map durch Schneetexturen ersetzt. Natürlich vorerst nur die Voll-Kacheln. Für die jeweiligen Übergänge müssen wir die Prozedur entsprechend oft wiederholen. Wer verhindern möchte, dass dieser Vorgang auf der ganzen Map angewendet wird, braucht nur die gewünschten Kacheln markieren, und dann auf „**Replace selected Cells**“ klicken. War nicht allzuschwer, hm ?

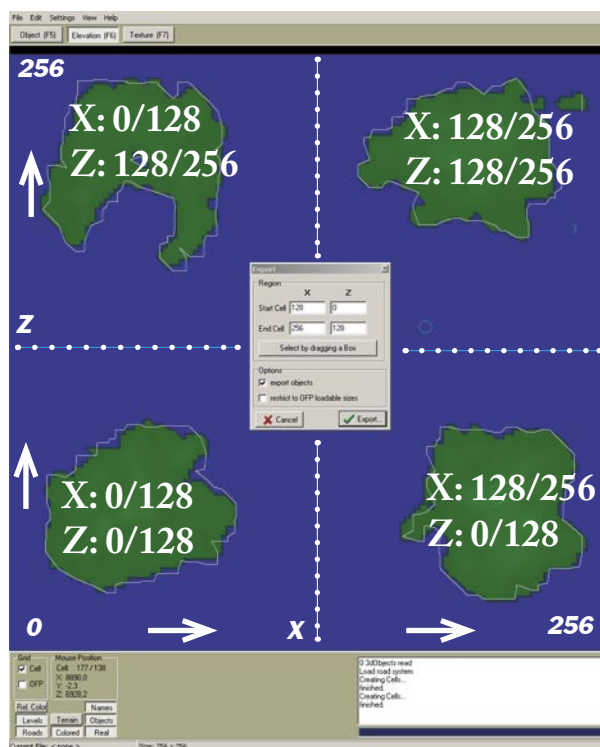


Ok, und genauso funktioniert es mit den **Objekten** – allerdings mit einem kleinen „Bonbon“ dazu! ;-)) Und zwar: wenn wir nur ein altes Objekt durch ein Neues ersetzen wollen, funktioniert es wie im Texture-Replacer. Doch zusätzlich haben wir hier die Möglichkeit, **mehrere** Objekte auf **einen Schlag** auszutauschen – einfach anwählen und mit „**Add from...**“ hinzufügen. Darüberhinaus können wir sogar mehrere neue Objekte in die Liste darunter aufnehmen – diese werden dann in ihrer Anzahl gleichmäßig verteilt. Wenn also z.B. 4 neue Objekte in der unteren Liste stehen, wird jedes einzelne Objekt zu 25% zugewiesen.

Zum Schluss noch ein kleiner Tipp, wenn es um das Auswählen alter, also schon platzierter Objekte/Texturen geht: einfach auf der Map den Zeiger über das Objekt/Textur bewegen, und dann „**Pos1**“ drücken. Das erspart das Umherschrollen in den Browsern. Das Austauschen von Objekten und Texturen macht sicherlich dann am meisten Sinn, wenn man eine schon bestehende Insel „Umskinen“ will. Also z.B. von Sommer nach Winter, von gemäßigt auf tropisch, usw.

Insel-Teile

Angenommen, jemand hat sich ein größeres Inselprojekt vorgenommen, und droht am Umfang des Projekts zu verzweifeln, oder eine kleine Gruppe möchte für eine Mission eine größere Insel realisieren, dann wäre es doch von Vorteil, wenn sich diese Aufgabe irgendwie aufteilen ließe?! Nun, dank WrpTool ist das kein Problem! Einer aus der Gruppe – sinnigerweise derjenige, welcher mit WT schon die meisten Erfahrungen gesammelt hat – kreiert zunächst eine neue Map mit der gewünschten Zellengröße, und erstellt schonmal eine/mehrere Inseln. Texturieren sollte er diese ebenfalls selbst, sofern die späteren Einzelteile (nehmen wir einfach mal an, die Map würde geviertelt) Festland durchtrennen würden. Ist das nicht der Fall (also 4 kleinere Inseln auf einer Map, wovon jede in einem 1/4-Quadranten Platz findet), kann ein jeder selbst seine Insel texturieren. Soweit so gut, jetzt kommt der spannende Teil: Die Insel wird jetzt „gevierteilt“, im wahrsten Sinne des Wortes. Dazu gehen wir bei „**File**“ auf „**Export WRP Cells...**“. Im neuen Fenster haben wir die Wahl zwischen Eingabemöglichkeit und mausorientierter Auswahl. Da bei mir das mit der Maus irgendwie nicht so funzt, beschränke ich mich auf die Eingabe – die ist sowieso genauer. Wir sehen vier Felder, in denen wir unsere Start- bzw. Endzellen unseres Inselteils definieren können. Dazu müssen wir wissen, welche Richtung mit „X“ bzw. „Z“ gemeint ist: „X“ beschreibt die Horizontale, und zwar von links nach rechts; „Z“ ist die Vertikale Achse, wobei „0“ unten ist. Im Bild rechts wird das System vielleicht deutlicher. So lassen sich ganz gezielt die jeweiligen Inselteile durch die Eintragung der Koordinaten an WT zum Export übergeben. Natürlich geht das immer nur für einen Teil auf einmal, die Prozedur muss also viermal durchgeführt werden. Haben dann die jeweiligen Mapper ihre Inselchen bearbeitet, kann sie der „Ober-Wraper“ ganz einfach wieder zusammenführen, indem er eine neue Datei mit der ursprünglichen Größe erstellt, und dann die Inselteile durch **importieren** zusammenführt. Dabei muss WT nur mitgeteilt werden, auf welches Viertel welche Insel gehört. Ganz nebenbei



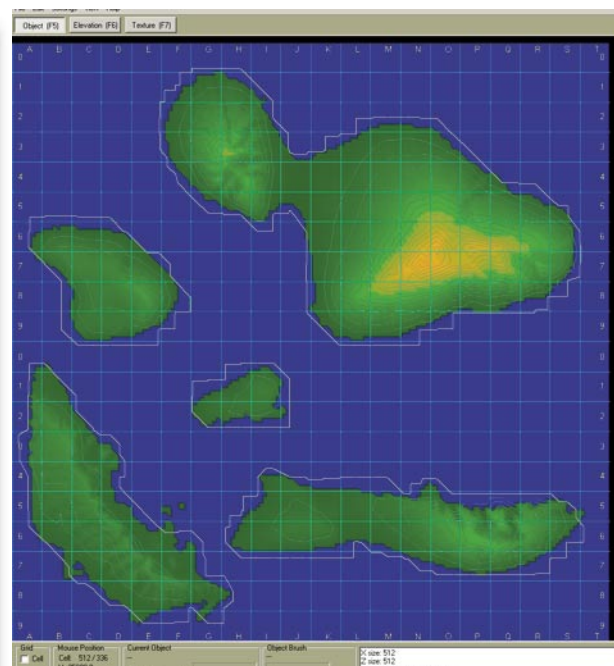
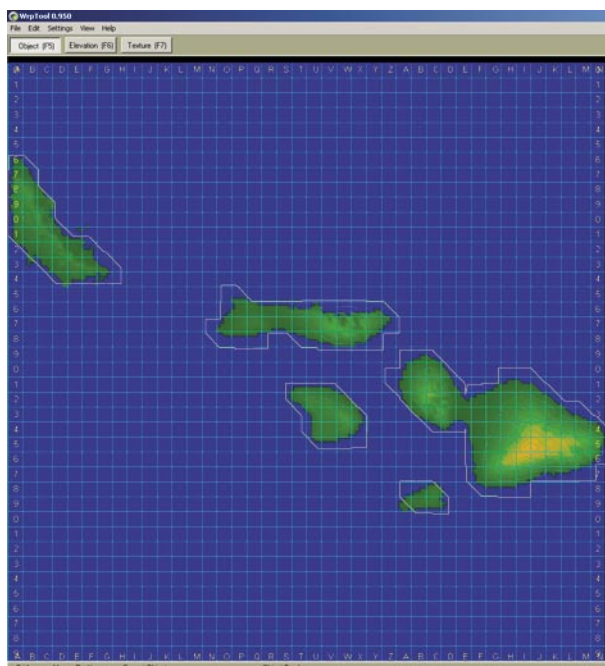
beinhaltet das Im-/Exportieren noch diverse Optionen, was alles beim Transfer „mitgenommen“ werden soll. Bei unserem ersten Beispiel wäre das wenig sinnvoll gewesen, aber ich nenne gerne eines, wo diese Optionen sehr nützlich sind: Angenommen, jemand möchte aus einer Auswahl schon fertiger Inseln mehrer auf einer einzigen Map haben – Null Problemo mit WrpTool! Und da kann ist es z.B. sinnvoll sein, nur das Terrain zu importieren, weil mancher vielleicht diese Inseln selbst mit Objekten versehen möchte, oder umgekehrt: Nur die Objekte sind das Ziel aller Begierde, das Terrain ist schon vorhanden, usw.usw. Die Fähigkeiten dieses Tools geben uns viele Möglichkeiten an die Hand, und enden meist erst beim Texturproblem – nämlich dass mit mehreren verschiedenen Inseln wahrscheinlich mehr als die erlaubten 512 verschiedener Texturen zusammengekommen sind. Eines noch: Wenn jemand nur einen Teil einer Insel direkt für OFP verfügbar machen möchte, muss er sich an die Regeln halten, die Anfangs schon erwähnt wurden: 2^2 ; 4^2 ; 8^2 ; 64^2 ; 128^2 ; 256^2 ; 512^2 und 1024^2 Zellen sind erlaubt! Wobei die drei letzten eher unsinnig sind – Logo. Jedenfalls ist man nicht gezwungen, einen Teilbereich aus einer der vier Ecken zu extrahieren (wie in unserem Beispiel), sondern man kann sich auch beliebige Teile aus einer Map „schneiden“, und wieder in eine neue Map „einpflanzen“. Doch vorsicht: Wenn ein neues Teilstück ein schon bestehendes beim Positionieren überlappen sollte, würden die betreffenden Teilbereiche einer vorher importierten Insel einfach überschrieben. Sollte dies geschehen sein, muss die Prozedur wiederholt werden – ein „Undo“ kennt WT nicht. Einzige Möglichkeit, dies zu vermeiden, ist häufiges Zwischenspeichern. Um ein weiteres Beispiel für die interessanten Möglichkeiten von WT anzuführen, schneide ich ein weiteres Thema kurz an, in welchem die eben besprochenen Funktionen ihre Nützlichkeit beweisen:

Real-Maps mit WT

R real-Maps? Kann man das essen? Nö, aber es ist genau das, was Res zu sein scheint – echte, richtige und reale Inseln in WT! Da staunt ihr, hm?! ;-) Nun, es dürfte sich herumgesprochen haben, dass die bekannten BIS-Inseln auch nichts anderes sind, als maßstabsgetreue Nachbildungen existierender Inseln – wenn ich mich richtig erinnere, waren die Halbinsel **Krk** vor ehem. Jugoslawien, sowie **Kreta** zwei der Vorbilder. Doch wie haben die das gemacht? Und wie zum Teufel kommen WIR dahin? Nicht ganz einfach, aber auch nicht soo schwer, dass der Aufwand nicht lohnen würde. Ich werde jetzt nicht ins Detail gehen, sondern nur auf das Prinzip eingehen, sodass jeder Interessierte schonmal die Richtung kennt.

Beginnen wir mit den wichtigsten Infos: Unsere Erde ist längst digital erfasst worden, Satellitentechnik macht's möglich (geht natürlich auch ohne, aber wäre viel zu teuer). Diese Daten liegen als sogenannte DEM (= Digital Elevation Model) im Internet bereit. Wir brauchen also nur noch die richtige Seite anzuschauen, und können uns von dort die nötigen Daten kostenlos herunterladen. Gebt einfach mal „[US Geological Survey Digital Elevation Models](#)“ in euren GOOGLE ein, und sucht dort nach „Hawaii“ auf der Karte, und klickt darauf. Ladet dann die Datei „**OahuH1.rar**“ herunter, und entpackt diese. Das dann sichtbare Dateiformat sollte euch bekannt vorkommen – richtig, es ist eine waschechte Terragen-Datei. Na prima, damit kennen wir uns ja schon aus, und können gleich zur Tat schreiten: Datei einfach mit **Wilbur** öffnen (nicht importieren), und dann wie gehabt wieder für WT abspeichern, in WT importieren und wie gewohnt weitermachen – fertig. Hey, jetzt haben wir eine echte, richtige Insel für OFP – wenn das nicht cool ist! ;-) Allerdings liegen die meisten Inseln als Fragmente bereit, man muss diese erst noch zusammenfügen. Dies geschieht mit einem eigenen Programm: „**3DEM**“ von [visualizationsoftware.com](#).

Ich habe dies mal mit der Inselgruppe „Maoi“ – ebenfalls bei Hawaii – praktiziert, und das Ergebnis unten links erhalten. Diesmal musste die Inselgruppe um die Hälfte verkleinert werden, da ich sonst WT „gesprengt“ hätte. Wie ihr seht, ist auf dieser 1024er Map reichlich Wasser vorhanden – das muss ja nicht sein. Ergo habe ich probiert, ob nicht alle Inseln auch auf einer handlicheren 512er Map Platz fänden. Dazu habe ich (wir kommen wieder zum Ausgangspunkt) einfach eine neue 512er Map erstellt, und Insel für Insel aus der großen Map extrahiert und neu auf der 512er positioniert. Wie ihr rechts unten seht, hatte das nach ein paar versuchen prima geklappt.



WT „frisieren“

Tja, nichts scheint unmöglich, liebe Wrapper! ;-) Wir können WrpTool noch ein gutes Stück an unsere persönlichen Bedürfnisse anpassen, sowie seinen Umfang noch ein gutes Stück erweitern. Allerdings gehe ich hier nur auf die Möglichkeiten als solche ein, ohne mich in Details zu verlieren – dazu reicht die Zeit einfach nicht.

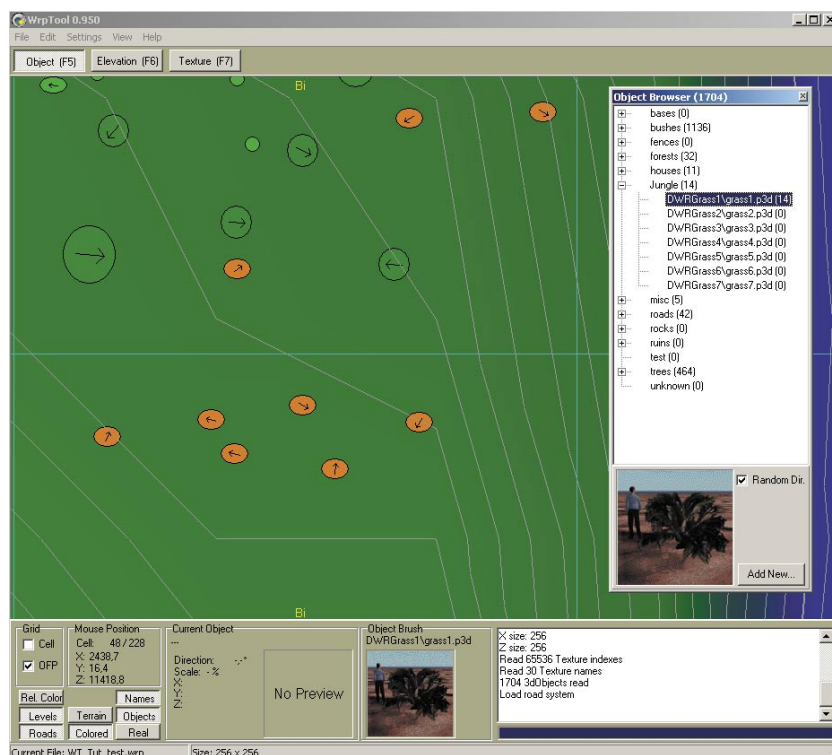
Beginnen wir mit den beiden schon erwähnten Möglichkeiten, neue Straßen und auch Wälder definieren zu können. Es wurden nicht alle Straßen und Wälder aus CWC und Res implementiert, man kann also auch ohne Addons noch weitere hinzufügen. Im Prinzip wird dabei WT nur mitgeteilt, welche neuen Element-Gruppen eine neue auswählbare Einheit bilden sollen. Bewerkstelligt wird dies mit den beiden „**Define**“-Tools im „**Settings**“-Menü. Beide Tools lassen sich nach ein paar Versuchen leicht bedienen und öffnen gerade mit **Addons** ganz neue Perspektiven.

Nicht so wichtig ist die Möglichkeit, den Punkt neu zu definieren, wann WT seine Darstellung von Schematisch auf Exakt (True Shape) umschaltet. Dies wird in der XML-Datei vorgenommen. Werden die Werte für „Simple LOD“ und/oder „Detail LOD“ (LOD= Level of Detail) vergrößert, behält WT länger eine detaillierte Darstellung bei beim herauszoomen. Hier ist die Hardware der entscheidende Faktor – wer die Werte vergrößert, dem knabbert WT mehr an den Ressourcen ;-)
Ebenfalls in der XML geschieht das Hinzufügen neuer Objekt-Kategorien, also z.B. Jungle-Objekte, diese erscheinen dann im Objektbrowser. Wurden dann darin neue Objekte aufgenommen, kann in der XML ein neues Symbol für die Simple-LOD-Darstellung hinzugefügt werden. Die Symbole können Rechtecke, Kreise und Ellipsen mit eigenen RGB-Farben sein – nützlich, die Addons sofort von den Originalobjekten zu unterscheiden.

Wer Addons hinzugefügt hat, oder auch von den Original-Objekten, der wird schnell etwas sehr unbefriedigendes feststellen: Es existiert keine **Vorschau** für die Objekte :-(Diesen Mangel darf man aber nicht den WT-Entwicklern anlasten, denn es ist eine Heidenarbeit, dies für alle möglichen Objekte zu realisieren. Doch ich möchte an dieser Stelle erneut dazu anregen, dass sich diese sicher sehr nützliche Arbeit auf mehrere Schultern verteilen sollte. Das Wie erkläre ich hier in Kürze:

Angenommen, man möchte aus einem **Addon** diverse neue Objekte seinem Objektbrowser hinzufügen, und dafür auch eine propere Vorschau generieren, dann gehen wir dazu folgendermaßen vor: Zuerst müssen die Objekte natürlich dem Objektbrowser hinzugefügt werden (**Add new**). Hernach suchen wir uns eine möglichst kleine und übersichtliche Insel – **Intro.wrp**, also die kleine Wüsteninsel aus OFP eignet sich vorzüglich – und laden diese in **OFP**. Im Editor platzieren wir die Objekte nebeneinander und steigen in die Vorschau ein. Stellen wir uns nun so vor das erste Objekt, dass es möglichst komplett auf dem Screen zu sehen ist. Vorteilhaft ist auch die Variante, dass man sich NEBEN das Objekt stellt (zum Größenvergleich bei Pflanzen grade zu unverzichtbar), und dann die Aussen-Kamera auf sich selbst und das Objekt richtet. Nach dem **Shot** per „**Print/Druck**“, schalten wir zu einem Bildbearbeitungsprogramm um, in welches wir den Screen laden. Damit wir ein Bild davon bekommen, wie der Screen abgespeichert werden muss, öffnen wir aus dem Ordner „**3DPreview**“ ein beliebiges Bild, und erforschen Bildgröße und Dateiformat, sowie Eigenschaften. Ein Preview-Bild muss als **.BMP** mit **128x128** Pixeln, sowie im **indizierten** Farbmodus vorliegen. Um die ganze Sache zu vereinfachen, erzeugen wir eine Kopie des Bildes, und haben damit auch dessen Eigenschaften für unseren eigenen Shot verfügbar. Wir brauchen also nur noch den Shot auf die richtige Größe umrechnen lassen (geht am schnellsten mit dem Freisteller, wenn dieser schon vorher auf das richtige Format 128x128x72dpi eingestellt ist). Nun ziehen wir das kleine Bild auf die **Kopie** aus 3DPreview, und speichern das Ganze mit folgenden Eigenschaften ab: Windows-**BMP**; **8 Bit** Farbtiefe (**indiziert**) und **RLE**-Komprimiert. Als Bezeichnung können wir vergeben, was wir wollen, aber Sinn macht sicher am ehesten, wenn diese sich möglichst an die der Objektbezeichnung anlehnt. Dieses Bild wird nun in den Ordner 3DPreview geschoben. Damit dieses nun aber vom Objektbrowser erkannt werden kann, muss es in der „**Objekts.ini**“ eingetragen werden. Dazu halten wir uns genau an die Regeln, wie wir sie bei den schon eingetragenen Objekten erkennen können: „**Pfad\Datei.p3d=Kategorie**“. Zusätzlich muss noch der Zusammenhang zwischen Objekt und Bild definiert werden. Dies geschieht mit der „**icon_images.ini**“ im Ordner 3DPreview. Wir geben wieder den Pfad und Objektnamen ein, und hinter das Gleichheitszeichen die Bildbezeichnung. Kleiner Tipp: in den meisten Fällen wird die Prozedur nicht aufs erste Mal klappen, und man bessert gefundene Fehler aus. Doch nach jeder Änderung einer **.ini** oder **.xml** muss WT neu gestartet werden, damit es die Änderungen initialisieren kann.

Wurde alles richtig gemacht, könnte das ganze wie rechts aussehen. Ich hatte dazu eine eigene Kategorie erstellt (siehe XML), und dort die sieben Palmen aus dem kleinen DWR-Addon eingefügt. Wie ihr sehen könnt, lassen sich meine orangenfarbigen Ellipsen leicht von den Original-Objekten unterscheiden, und



überdies existiert die gewünschte Vorschau – durchaus nützlich also, sich sein WT ein wenig zurechtzubiegen! :-)

Aber wie schon gesagt: es ist schon ein gutes Stück arbeit, dies mit einer Vielzahl von Addons durchzuführen. Daher mein Aufruf an alle, die sich selbst eine Vorschau basteln: bitte veröffentlicht diese für alle anderen. Es wird zu diesem Zweck auf Nekromantix.com ein Pool eingerichtet werden, auf den dann alle Wrapper zugreifen können. Mit der Zeit sind dann bestimmt für die gebräuchlichsten Addons eigene Previews vorhanden, und alle sind glücklich! ;-) Das Wie sollte ganz einfach sein: Es werden nur die Bilder, sowie eine simple Textdatei benötigt. In die Textdatei werden die Angaben aus der *Objekts.ini*, der *icon_images.ini*, sowie ggf. aus der *Objekts.xml* hineinkopiert – fertig. Natürlich nur der selbsterstellte Teil, und nicht alles was drin ist. Am Schluss wird das Paket gezippt, und inklusive einer Aufzählung der Objekte, für die eine Preview erzeugt wurde, an flashpoint@nekromantix.com gemailt. Aber bitte überzeugt euch VORHER, ob auch wirklich alles so funktioniert, wie es vorgesehen ist. Ich glaube nicht, dass irgendjemand die Zeit hat, alle Einsendungen vorher zu kontrollieren.

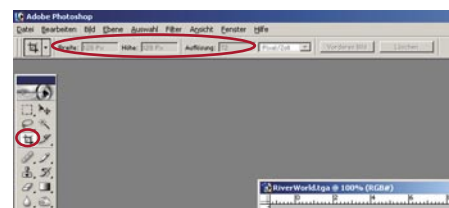
Es gibt sicher noch die eine oder andere Sache, die ich in diesem Tut nicht angeschnitten habe, doch die wichtigsten Dinge sollten enthalten sein. Es fehlt nur noch ein wichtiger Schritt: Wie bekomme ich meine Insel mit allem Drum und Dran für OFP gebacken und aufbereitet, und damit auch bereit für andere Spieler? Der nächste Artikel wird auch dies erläutern.

WrpTool > OFP

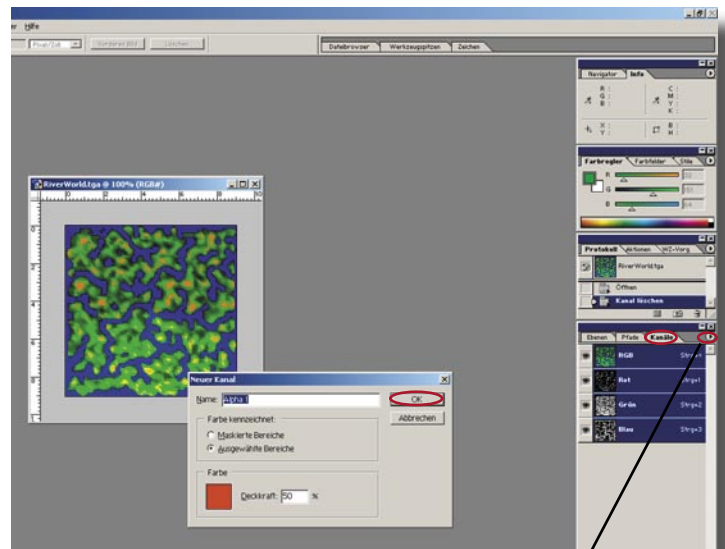
Uffz! Wir haben uns bis hierher durchgehangelt, und atmen noch ...wenn auch nur noch flach! ;-) Also, auf zum letzten Gefecht, die Insel wird jetzt für OFP zurechtgeschniegelt.

Zunächst ein paar grundlegende Anmerkungen: Ich gehe davon aus, dass sich kein totaler Newbe unter euch befindet, was Editieren unter OFP betrifft – alles total von Grund auf zu erklären, würde diesen Rahmen sprengen. Zudem bin auch ich kein absoluter Profi, insbesondere was das Scripten – hier unsere „config.cpp“ – betrifft. Aber ich denke, wir halten uns einfach an das unbedingt notwendige, dann klappt das schon. Alles, was dann darüber hinausgeht – ich denke da z.B. an die Sounds, ILS-Traffic, Ortsnamen samt Schildern, usw. – wird in diesem Tut einfach ausgeblendet – zumindest vorerst. Früher oder später, wenn ich mit dieser Problematik thematisch durch bin, wird es eben ein Update auf dieses Tut geben, das dann auch diese offenen Punkte eingehend behandelt. Im Moment werde ich mein Bestes geben, damit ihr auf jeden Fall euer neues Wunderland in OFP gebacken kriegt.

Beginnen wir also zuerst mit der Erstellung des kleinen Vorschaubildes, das wir im OFP-Editor sehen können. Bisher war das ja – behelfsmäßig – das „Dessert-Island“, aber um andere nicht zu verwirren, und selbst den Überblick zu behalten, kreieren wir unser eigenes Vorschaubild. Dazu gehen wir wieder in WrpTool, und laden unsere neue Insel. Zoomt jetzt soweit heraus, dass die Insel ganz im Hauptfenster zu sehen ist – Texturen und Höhenlinien könnt ihr abschalten. Machen wir davon mit der „**Druck**“-Taste einen Screenshot, und fügen diesen in einem Bildbearbeitungsprogramm ein – in diesem Fall **Photoshop** (PainShopPro-User lesen im Original-Tut von WrpTool nach). Da wir nun eine neue Ebene erhalten haben, reduzieren wir diese wieder auf die Hintergrundebene (obere Ebene muss aktiv sein, dann „**Strg**+“**E**“ drücken). Als nächstes schneiden wir den Teil mit der Insel heraus, und verändern die Bildgröße auf folgende Parameter: 64x64; 128x128 oder 256x256 Pixel bei 72dpi Auflösung. Das geht auch ganz gut mit dem Freisteller schon beim Ausschneiden – einfach oben links die Pixelwerte und die Auflösung eingeben, dann den Freisteller auf den gewünschten Ausschnitt aufziehen, „**Enter**“ drücken – fertig. Fügen wir als nächstes einen leeren **Alpa-Kanal** zum Bild hinzu. Den braucht OFP, auch wenn dieser hier



keine Funktion erfüllt. Dazu gehen wir zum Fenster „**Kanäle**“, und fügen dort einen neuen Alphakanal ein. Das geht aber nicht mit dem Symbol unten in der Tool-Leiste, sondern nur über das kleine schwarze **Dreieck** rechts oben am Kanal-Fenster! Sobald der neue Kanal im Kanalfenster erschienen ist, können wir auch schon speichern. Das Bild muss als xxx.**tga**-File mit **24Bit** Farbtiefe, und ohne Komprimierung gespeichert werden. Das Häkchen bei „Alpha-Kanäle“ im Speicherndialog muss natürlich gesetzt sein. Als Bildname sollte derselbe wie für die Insel gewählt werden. Um ganz sicher zu gehen, sollten wir das eben gespeicherte Bild nun schließen, und gleich wieder öffnen. Ist der Alpha-Kanal tatsächlich vorhanden? Ja? Gut, dann weiter.



Hier neuen Alpha anlegen

Damit unser xxx.tga-File nun für OFP lesbar wird, müssen wir es in ein xxx.**paa**-File umwandeln. Das geschieht mit „**Texview**“ von BIS. Öffnen wir also unser xxx.tga in Texview, und speichern es als xxx.**paa** ab. Sollten wir nach dem Öffnen nur ein leeres Feld gesehen haben, ist beim Abspeichern in Photoshop etwas fehlgeschlagen. Ansonsten war's das schon mit Texview.

Öffnen wir als nächstes eine entsprechende **Config.cpp** aus dem **WrpTool/Example_Configs/wrptool_xxkm/-**Ordner. Das „xx“ steht für die verwendete Inselgröße. In der Config.cpp ersetzt ihr nun alle „wrptool_xxkm“-Einträge durch euren Inselnamen, und speichert die Config.cpp neu in einen Ordner, der ebenfalls euren Inselnamen trägt. Dorthinein kommt jetzt auch das xxx.paa-Bild. Falls jemand eine kleinere Insel als 256 Zellen erstellt hat, der nimmt einfach die jeweiligen Dateien der kleinsten Insel, also „wrptool_12km“ – funktionieren wird es trotzdem.

Fehlt vorerst nur noch das Packen des Ordners in als xxx.**pbo**-File. Ich hatte es zunächst mit „**StuffPBO**“ versucht, jedoch nur eine Fehlermeldung erhalten. Wenn es bei euch funzt, nehmt dieses Programm. Falls ebenfalls nicht, bei mir hatte es dann mit „**PBO Tool**“ funktioniert. Dieser Packer wird mehrfach in der OFP-Community zum DL angeboten, und vielleicht habt ihr ihn ja schon. Nennt die gepackte Datei ebenfalls wie eure Insel, aber das sollte auch so klar sein. Jetzt könnt ihr die PBO-Datei mit der

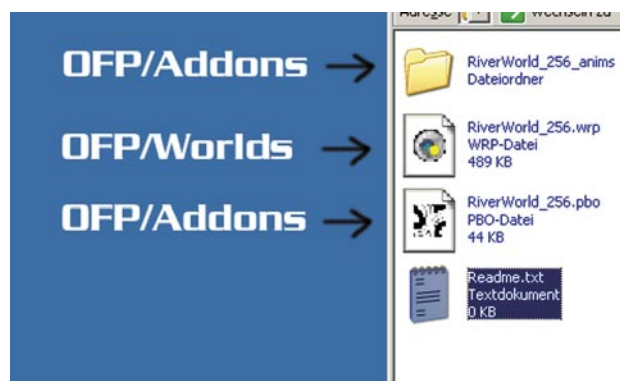
Config.cpp und dem xxx.**paa**-Bild in OFP/Res/**Addons**, sowie eure xxx.**wrp**-Inseldatei in OFP/**Worlds** kopieren. Das wars schon fast, fehlt nur noch die **Animation** für die Insel.

Dazu verwenden wir erstmal eine der mitgelieferten **Anims** aus dem **WrpTool/Example_Configs/OFP/Addons**-Ordner mit der entsprechenden Größe, und kopieren ihn in die OFP/**Addons**-Folder. Damit die Animation von der config.cpp angesprochen werden kann, müssen wir ihr den Namen unserer Insel verpassen, den wir auch in der config.cpp verwendet hatten. Das sollte dann ungefähr so aussehen:

OFP/Addons/MyWorld_128_Anims/Intro.Myworld_128

MyWorld_xxx steht dabei für eure Insel und deren Größe. Die Größenangabe ist nicht zwingend, es ist halt von Vorteil, wie Eingangs schon erwähnt wurde.

Wir haben die ganze Konfiguration jetzt abgeschlossen, und dabei so angelegt, dass ihr immer noch problemlos Änderungen durchführen könnt. Sollte die Insel aber das finale Stadium erreicht haben, und Anderen zur Verfügung gestellt werden, können wir die xxx.**wrp**-Datei noch mit „**Binarize**“ komprimieren. Sie wird damit erheblich kleiner, und der „**Modem-Sauger**“ freut sich. Ganz am Schluss sammeln wir noch alle Ordner und Dateien in einen gemeinsamen Ordner, in den wir auch eine ausführliche Readme beifügen.



Nun mag sich der eine oder andere fragen, ob man denn nicht die ganzen Texturen und xxx.p3d-Files irgendwie mit hineinpacken muss?! **Bloß nicht!** Bisher haben wir ja nur Dateien verwendet, die OFP alle schon mitgebracht hat. Selbst wenn wir (später) inoffizielle Addons auf unserer Insel verwenden, genügt es vollkommen, die verwendeten Addons in der Readme auszuweisen. Der User muss dann nur das betreffende Addon herunterladen, und in den Addon-Ordner legen – schon funzt wieder alles. Aber keine Regel ohne Ausnahmen: Wenn jemand von meinen „Jack“-Texturen nur ganz wenige verwendet hat – z.B. für ein winziges Eiland mit nur zwei Texturen+Transitions – kann diese Ausnahmsweise mit hineinpacken. Aber dies muss dann ausdrücklich in der Readme ausgewiesen sein, denn wenn jemand das „Jack“-Set schon haben sollte, muss er dieses dann deaktivieren, damit es nicht zu Problemen kommt.

Mir ist bewusst, dass der letzte Teil für den Laien wahrscheinlich sehr verwirrend sein muss – ich kann das bestens nachvollziehen, auch ich muss jedesmal wieder überlegen: „...mensch, wie war das gleich nochmal?!“ Man tut sich wirklich am leichtesten, wenn man hier so viel wie möglich bei schon funktionierenden Inseln „nachspickt“, und dann möglichst die Beispiele aus der „Example_Config“ für seine Bedürfnisse „zurechtbiegt“. Im Lauf der Zeit bekommt man dann immer mehr Sicherheit, und man beginnt zu begreifen, wie das Ganze funktioniert. Auch hier gilt wieder: Übung macht den Meister, denn der fällt nicht vom Himmel. Hier lernt man sicher am meisten durch Fehler, die man am Anfang fast zwangsläufig macht. Und auch hier verweise ich dringend darauf, immer wieder mal einen Blick in das beiliegende Manual zu werfen, denn auch dort wird gerade der letzte Schritt eingehend beschrieben, inklusive einiger Beispiel-config.cpp. Und auf keinen Fall aufgeben am Schluss! Wenn ihr wirklich nicht weiterkommt, dann seht in den Foren der OFP-Community nach, denn dort trifft ihr mit Sicherheit auf User mit den gleichen Problemen. Zwei Anlaufstellen empfehle ich besonders: Zum einen das AEF-Forum, dort erhaltet ihr die Unterstützung der Programmentwickler selbst (**Dschulle** und Co), sowie das **Mapfact**-Forum, wo ich mich um eure Probleme kümmern werde, so gut ich kann. Aber ich bin fest davon überzeugt, dass ihr es schaffen werdet, eure neuen Welten zu realisieren, denn WrpTool ist das bisher am leichtesten und besten zu bedienende Programm, das mir bekannt ist.



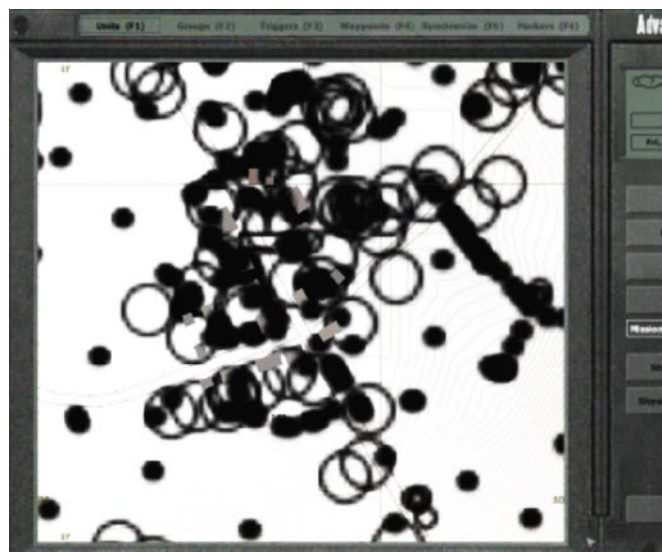
„Seltsamkeiten“

Diese nachfolgend besprochenen „Eigenheiten“, die mit manchen Inseln in OFP einhergehen können, sind weniger WT anzulasten, als vielmehr der Tatsache, dass WT einfach vielmehr Möglichkeiten offeriert, als es in OFP jemals vorgesehen war. Wer sich in etwa an den ursprünglichen Rahmen-Spezifikationen für OFP orientiert, wird von den nachfolgend besprochenen Effekten wenig oder gar nichts bemerken. Doch besprechen wir im Einzelnen, was gerade bei größeren Inseln passieren könnte:

Es können z.B. ungewohnte „Zitter“-Effekte auftreten, die sowohl im Cockpit, als auch in der normalen Infantry-Perspektive beim Waffenwechsel zu sehen sein können.

Problematisch können große Inseln, oder auch normal große Inseln mit viel Objekten, für die KI werden, wenn die Wegpunktdistanzen zu groß werden. Die KI steht dann ein bissl in der Gegend herum, und „verliert die Lust“ nach weit entfernten Wegpunkten zu suchen. Dies kann allerdings auch auf den Original-Inseln passieren. Abhilfe lässt sich durch mehrerer kurze Wegpunktentfernungen statt eines weit entfernten Wegpunktes schaffen.

Ein weiterer Effekt, der bei großen Inseln auftritt, ist die Symbol-Darstellung im OFP-Editor. Die Zeichen für Vegetation werden übertrieben groß dargestellt, und überdecken einander, sowie andere Objekte. Das ist wenig übersichtlich, lässt sich jedoch laut WT-Team in der „Resource.bin“-Datei beheben. Allerdings müsste dann die Insel in einen eigenen MOD-Ordner, oder die Insel ist Teil eines anderen Addons. Leider lassen sich die Entwickler nicht weiter über dieses Thema aus, und ich hatte auch noch keine Gelegenheit, mich damit zu beschäftigen. Ich hoffe aber, dass sich in den einschlägigen Foren früher oder später eine praktikable und verständliche Lösung finden lässt. Bis dahin müssen wir damit leben, dass dieser Effekt bei Inseln $>512^2$ auftritt.



Es war einmal...

...ein Plan. Ja, ich hatte einen – ganz ehrlich! Doch wie das nicht selten mit Plänen so ist, entpuppen diese sich lediglich als gute Absichten! ;-) Ich hab doch glatt gedacht, ich könne meine zeitlichen Versprechen bei euch einlösen – doch leider war die Realität ganz anderer Meinung. Nicht nur, dass WrpTool noch vor Fertigstellung meines Tutorials eine umfangreiche Neuauflage erfahren hatte, auch mein Leben musste sich einem „Update“ unterziehen. Und so kam's, dass ich euch einen Mischaufluss servieren musste, um wenigstens in absehbarer Zeit eine akzeptable Näherung des gewünschten Ergebnisses liefern zu können.

Die Versionsnummer „v.0.9“ deutet also schon an, dass dieses Tut beileibe nicht der Weisheit letzter Schluss ist. Nicht, dass ich mir einbilde, mit Mr-Murrays Script-Bibel mithalten zu können, aber immerhin ist das Die Latte, die ich mir gesteckt hatte. Doch bevor ich mir weitere Gedanken über eine weitere, und dann hoffentlich bessere Version mache, möchte ich eure Reaktionen auf mein Tut abwarten. Nur so kann ich eure Wünsche und Anregungen in eine Überarbeitung einfließen lassen. Primär möchte ich erstmal beobachten, was für Fragen im Mapfact-Forum am häufigsten gestellt werden, wo euch der Schuh also am meisten drückt. Diese Beobachtungen werden eine Zeit lang anhalten, bis sich ein klares Bild kristallisiert, und ich einen neuen Plan schmieden kann.

Doch zeitgleich muss und möchte ich selbst erstmal mehr praktische Erfahrungen mit WT sammeln. Ferner sollte ich auch meinen Aufgaben als Betatester für WT nachkommen, schließlich wird WT ständig weiterentwickelt und gepflegt. So kann ich eure Wünsche und Erfahrungen – sofern sinnvoll und realisierbar – an das Entwicklerteam weiterleiten.

Ein Wort noch zum „Update“: Damit war gemeint, dass ich seit Anfang dieses Jahres als Freelancer tätig bin, und somit leider nicht mehr in der Lage bin, über feste Freizeiten zu verfügen. Nach diesem Release des Tuts können in naher Zukunft also allenfalls kosmetische Korrekturen, sowie natürlich Fehlerbeseitigungen realisiert werden. Eine „v.1.x“ wird also aus mehreren Gründen sicher noch einige Zeit auf sich warten lassen müssen. Dafür bitte ich um euer Verständnis, und bedanke mich ganz herzlich für die Geduld und das Interesse, das ihr bis hierher schon bewiesen hattet.

Ich wünsche euch noch viel Spaß und Erfolg beim Wrappen!

Viele Grüße, euer „Jack“