

WA-Workshop

Voraussetzungen

- [Tiled \(https://www.mapeditor.org\)](https://www.mapeditor.org) installiert
- folgende TileSets heruntergeladen:
 - [cija_expansion \(ByteWerk\)](https://github.com/cijavel/tileset_Bytewerk/blob/main/cija_32x32_expansion%20)
(https://github.com/cijavel/tileset_Bytewerk/blob/main/cija_32x32_expansion%20)
 - [mapUtilities \(c3CERT\)](https://github.com/c3CERT/rc3_tiles/blob/master/imgs/tilesheets/mapUtilities.pr)
(https://github.com/c3CERT/rc3_tiles/blob/master/imgs/tilesheets/mapUtilities.pr)
 - [misc \(c3CERT\)](https://github.com/c3CERT/rc3_tiles/blob/master/imgs/tilesheets/misc.png)
(https://github.com/c3CERT/rc3_tiles/blob/master/imgs/tilesheets/misc.png)

Grundlagen

- logische Struktur der Welt

```
digraph hierarchy {  
    nodesep=1.0  
    node [color=Red,fontname=Arial,shape=box] //All nodes will this  
shape and colour  
    edge [color=Blue, style=dashed] //All the lines look like this  
  
    World->{Map1 Map2 MapN}  
    Map1->{Layer1 Layer2 LayerN}  
    Layer1->{Tile1 Tile2 TileN}  
}
```

- Tiles sind 32x32px
 - mehrere Tiles in TileSets zusammengefasst
- Layer fügt Tiles in zwei Dimensionen zusammen
 - Rastergröße gleich Tile-Größe (also zwei Tiles auf einem Layer können sich nicht überschneiden)
- Map fügt Layer zusammen
 - Layer überdecken sich "von unten nach oben"
 - Layer können zueinander versetzt sein - deren Raster müssen sich also nicht überlagern
- für die Welt müssen Karten (im JSON-Format) und TileSets verfügbar sein
- (Inter)aktionen können beim Betreten von Tiles ausgelöst werden
 - Aktion über Eigenschaft des Layers definiert

Vorbereitung

- Verzeichnisstruktur:
 - project/
 - [awesomeP].tilted-project
 - maps/
 - **main.json**
 - other.json
 - cija_32x32_expansion for Pipoya_CC0.png
 - ...
- in Tiled
 - *Save Project As:* im Ordner project
 - *Add Folder to Project:* maps
 - *View -> Show Tile Animations*
 - *View -> Highlight Current Layer*

Maps bauen

jede Map

- braucht einen Tile-Layer namens *start*
 - "Interaktion" spawnen
- braucht einen Object-Layer namens *floorLayer*
 - hier werden die Avatare gezeichnet

neue Map anlegen und TileSet verknüpfen

- neue Map

New Map

Map

Orientation: Orthogonal

Tile layer format: CSV

Tile render order: Right Down

Map size

☐ Fixed

Width: 100 tiles

Height: 100 tiles

3200 x 3200 pixels

☒ Infinite

Tile size

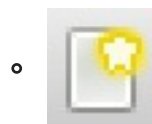
Width: 32 px

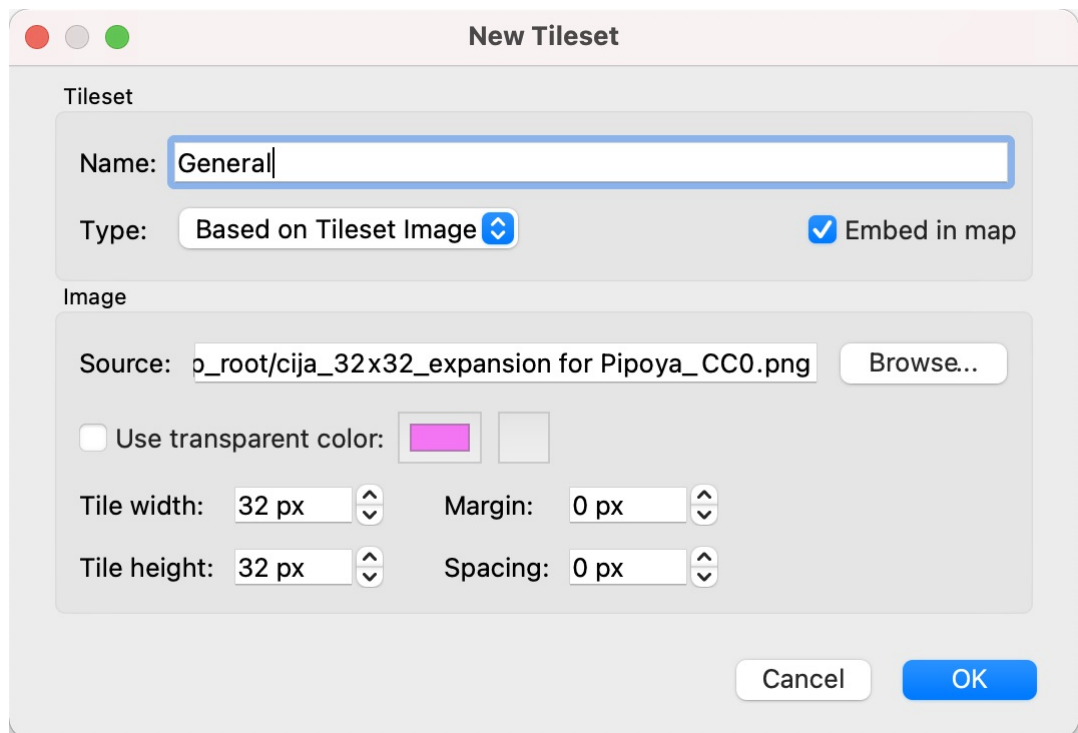
Height: 32 px

Cancel Save As...

- Speichern als *JSON map files (*.json)*






- Tileset einbinden





- Tile-Größe prüfen
- [x] bei *Embed in map* notwendig

Tiles setzen

Tool	Aktion
	setzt die jeweils aktuell ausgewählten Tiles einmalig auf den Layer
	setzt die jeweils aktuell ausgewählten Tiles mehrfach auf den Layer (füllt das mit dem Tool gezogene Rechteck mit der Tile-Selektion auf)
	Löscht Tiles
	wählt Tiles auf dem Layer aus
	Verschiebt das Raster des aktuellen Layers

Properties der Map


- Map -> Map Properties...
 - *Infinite* muss vor dem Bereitstellen der Map wü die Welt entfernt werden

Custom-Properties eines Layers

- Layer muss dazu ausgewählt sein


Name	Typ	Beschreibung
startLayer	bool	gesetzte Tiles werden als Spawn-Punkte behandelt
exitSceneUrl	string	betreten gesetzter Tiles führt zum Sprung in eine andere Map
openWebsite	string	betreten gesetzter Tiles führt zum Öffnen der Website im Overlay
silent	bool	gesetzte Tiles unterbinden P2P-Jitsi
jitsiRoom	string	betreten gesetzter Tiles öffnet Jitsi-Raum

Custom-Properties eines Tiles

- 
 - kann auf einer beliebigen Selektion von Tiles gesetzt werden

Name	Typ	Beschreibung
collides	bool	Avatar wird blockiert

Animationen

- 
 - Tile auswählen, die die Animation beherbergen soll (kann eine leere Tile sein), dann



- Animations-Tiles durch Doppelklick hinzufügen

Zwischen Maps springen

- die Property `exitSceneUrl` enthält den Pfad der Map, auf die gesprungen werden soll, sowie ggf. den Spawn-Layer als URL-Fragment (der Teil hinter #)
- Beispiel:
 - **other.json** springt auf die Map `other.json` und spawnt auf einem gesetzten Tile des Layers `start`
 - **other.json#hidden** springt auf die Map `other.json` und spawnt auf einem gesetzten Tile des Layers `hidden` (**Achtung:** der Layer muss die Custom-Property `startLayer` gesetzt haben)

Testen

- Die Test-Infrastruktur vom rc3 ist noch aktiv
- die Karten-Daten (alles im Ordner *maps*) müssen über https abrufbar sein
 - bei c4E musste der Webserver auch einen CORS-Header zurückliefern:
Access-Control-Allow-Origin `https://test.visit.at.wa-test.rc3.cccv.de`

- Beispiel:
 - Karte liegt unter <https://example.com/maps/main.json>
 - dann ist die Test-URL https://test.visit.at.wa-test.rc3.cccv.de/_global/example.com/maps/main.json
 - der Browser-Cache ist hier der Feind
- s. <https://howto.rc3.world/maps.html#testen>

Links

- [Tiled-Homepage \(https://www.mapeditor.org\)](https://www.mapeditor.org)
- [TileSet-Repository des rC3 \(https://git.cccv.de/rc3/world-tiles\)](https://git.cccv.de/rc3/world-tiles)
- [rC3-World HowTo \(https://howto.rc3.world/maps.html\)](https://howto.rc3.world/maps.html)