City of Gangsters: Map Modding Part 2

Updated Sept 23 2021

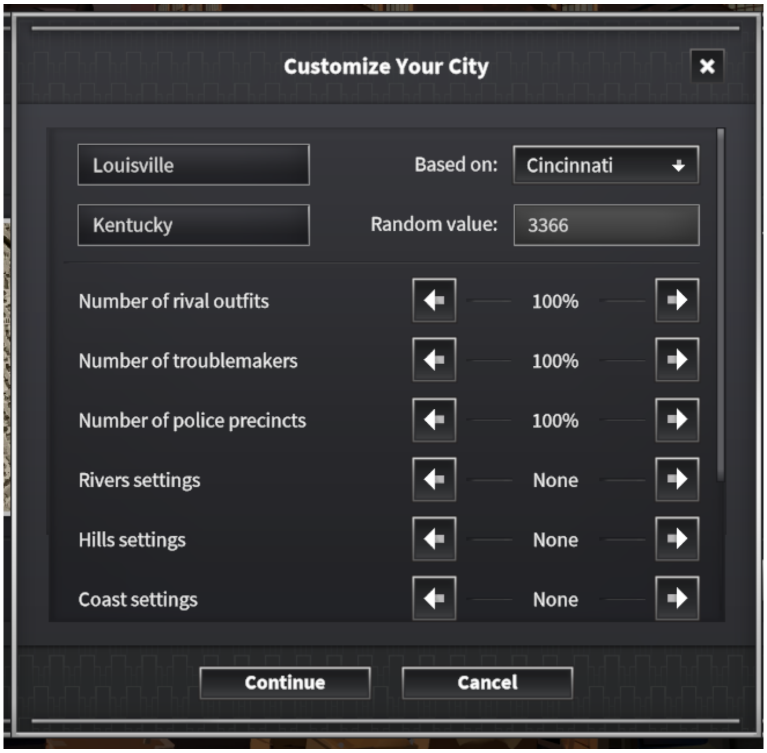
Previously in Part 1, we created a new map file and uploaded it to Steam. In this Part 2, we will make a new map that we will customize.

Generating a blank city

Let’s make our own version of the city of Louisville, Kentucky.

Start a new game with the “Custom City” option, and overwrite the city name and state name to say “Louisville” and “Kentucky”.

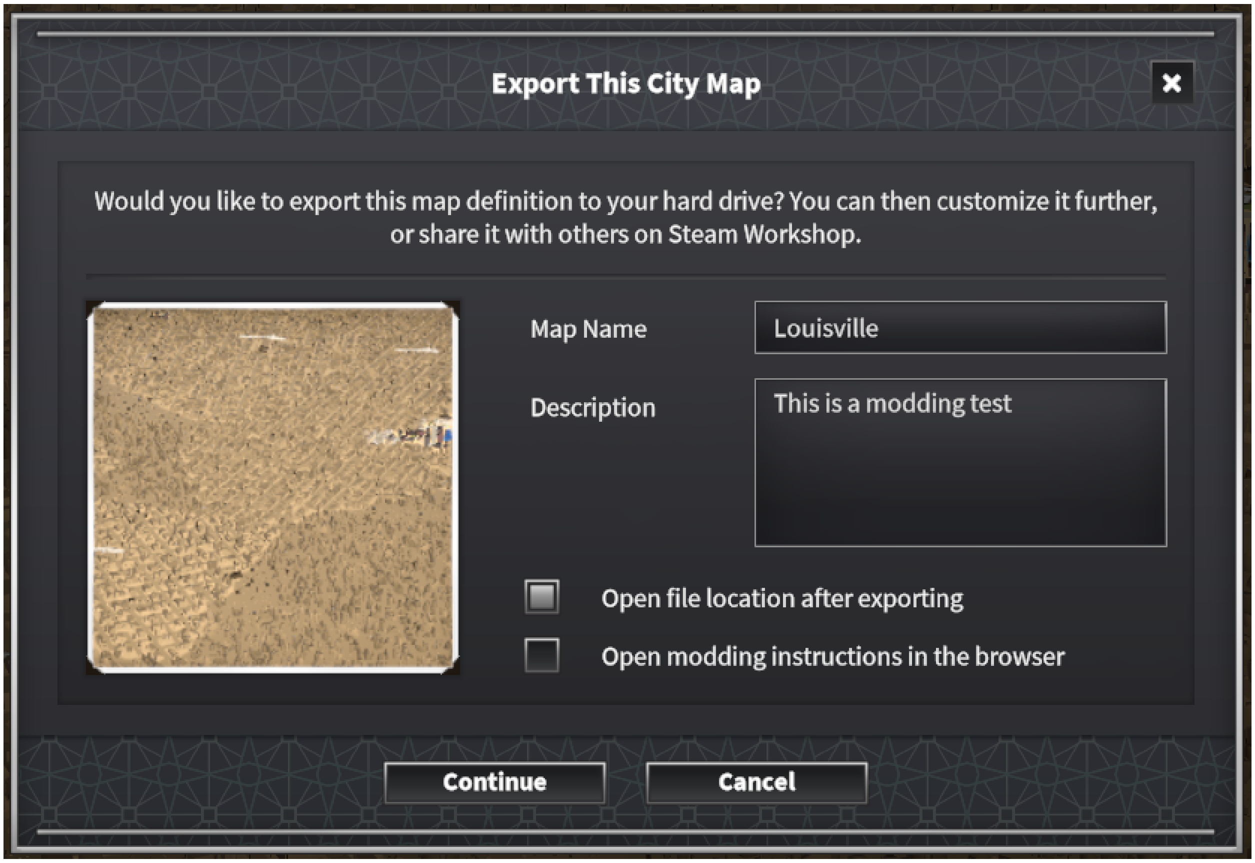
Then set the parameters as follows - it’s important that rivers, hills, and coasts be set to “none”, the rest is up to you. For the “based on” starting city, pick Cincinnati if you have the Deluxe edition, or if not, pick Detroit:



Louisville
Based on: Cincinnati
Kentucky
Random value: 3366

Number of rival outfits 100%
Number of troublemakers 100%
Number of police precincts 100%
Rivers settings None
Hills settings None
Coast settings None

Start the game, which should produce a flat map, and export it as a mod, as described in Part 1:



Export This City Map

Would you like to export this map definition to your hard drive? You can then customize it further, or share it with others on Steam Workshop.

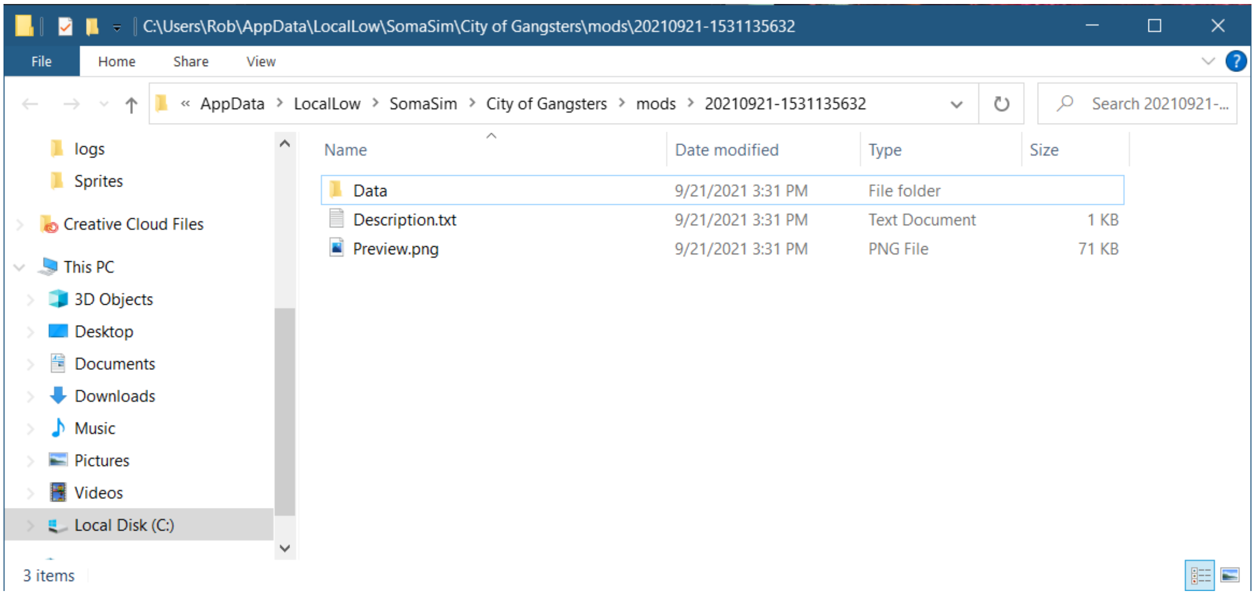
Map Name: Louisville

Description: This is a modding test

☑ Open file location after exporting

□ Open modding instructions in the browser

This should produce a new folder:



C:\Users\Rob\AppData\LocalLow\SomaSim\City of Gangsters\mods\20210921-1531135632

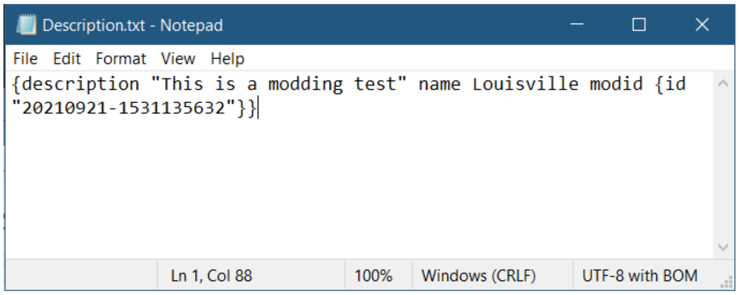
« AppData › LocalLow › SomaSim › City of Gangsters › mods › 20210921-1531135632

Name	Date modified	Type	Size
Data	9/21/2021 3:31 PM	File folder	
Description.txt	9/21/2021 3:31 PM	Text Document	1 KB
Preview.png	9/21/2021 3:31 PM	PNG File	71 KB

3 items

Each mod is a directory that always has the same files and structure:

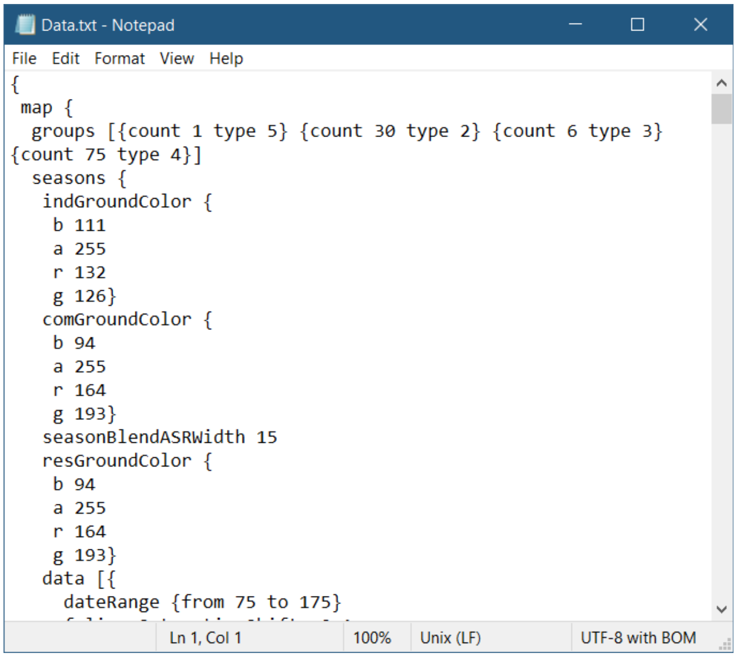
- **Preview.png** - this is the image that shows up on the Workshop page
- **Description.txt** - name and description that shows up both in Workshop and in the game, for example:



```
{description "This is a modding test" name Louisville modid {id
"20210921-1531135632"}}
```

- **Data** folder contains **Data.txt** which is the map definition we're going to edit.

If you open Data.txt, it might look something like the following:



```
{
 map {
  groups [{count 1 type 5} {count 30 type 2} {count 6 type 3}
{count 75 type 4}]
  seasons {
   indGroundColor {
    b 111
    a 255
    r 132
    g 126}
   comGroundColor {
    b 94
    a 255
    r 164
    g 193}
   seasonBlendASRWidth 15
   resGroundColor {
    b 94
    a 255
    r 164
    g 193}
   data [{
     dateRange {from 75 to 175}
```

d

Most of this data is not going to be useful, but look for the following sections:

- **mapsize** - defines size of the map in game units (e.g. 400x400). For reference, a single-family building is one unit wide, and a city block is typically 10 units long (say, 8 houses plus a road which is two units wide), with some variation.
- **waternodes** - this defines segments of water that make up rivers and lakes
- **mountainnodes** - defines hills (they're not tall enough to be mountains, honestly ;))
- **districts** - defines named neighborhoods that show up on the map

We'll learn how to mod them in that order.

Adding a river

Map size cannot be changed, but we will use it as a guide for placing everything else.

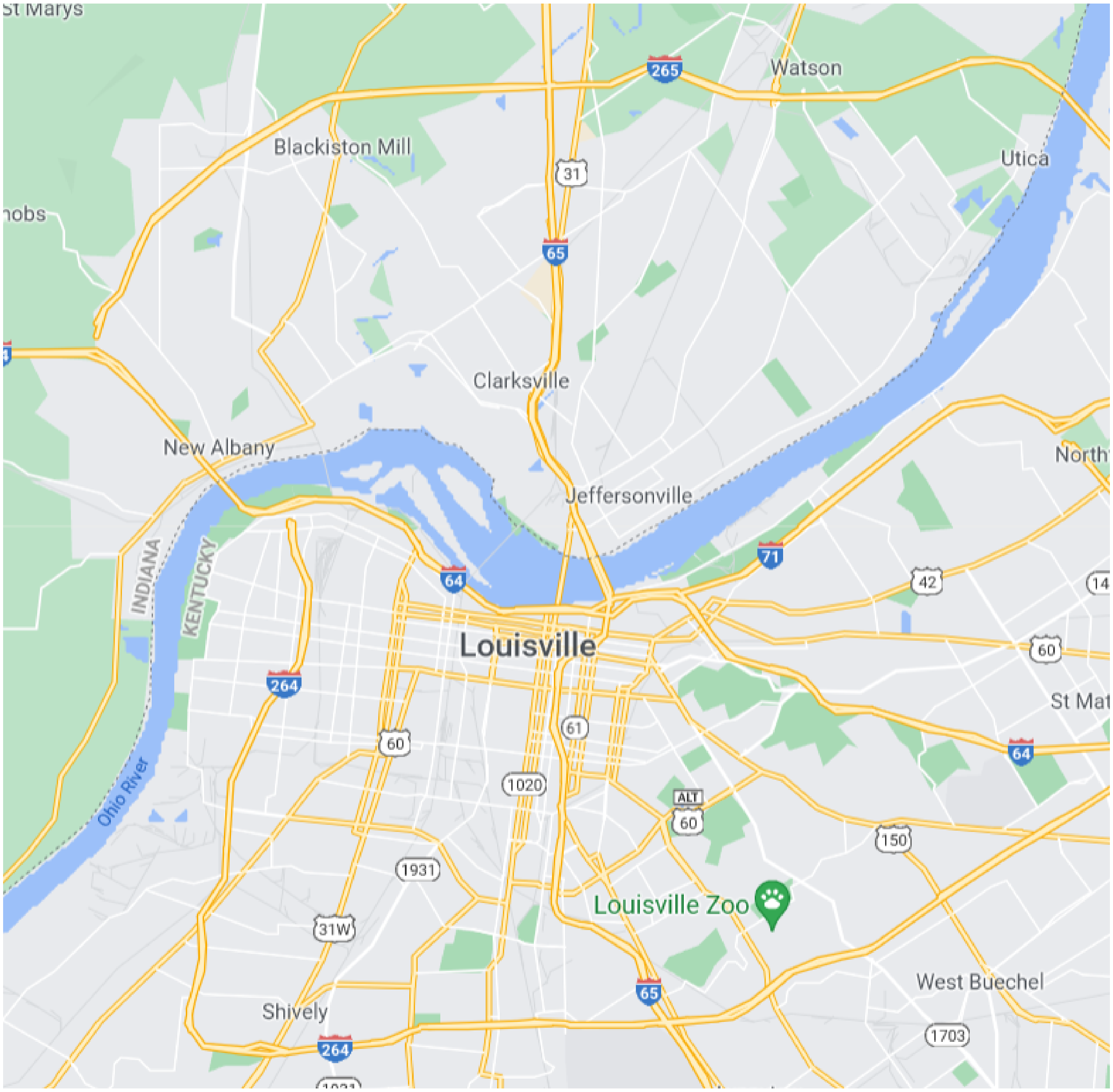
Find the mapsize in your map file. If you also started from Cincinnati, it should say:

```
mapSize { width 400 height 400 }
```

Also, here are the map sizes for the four started cities:

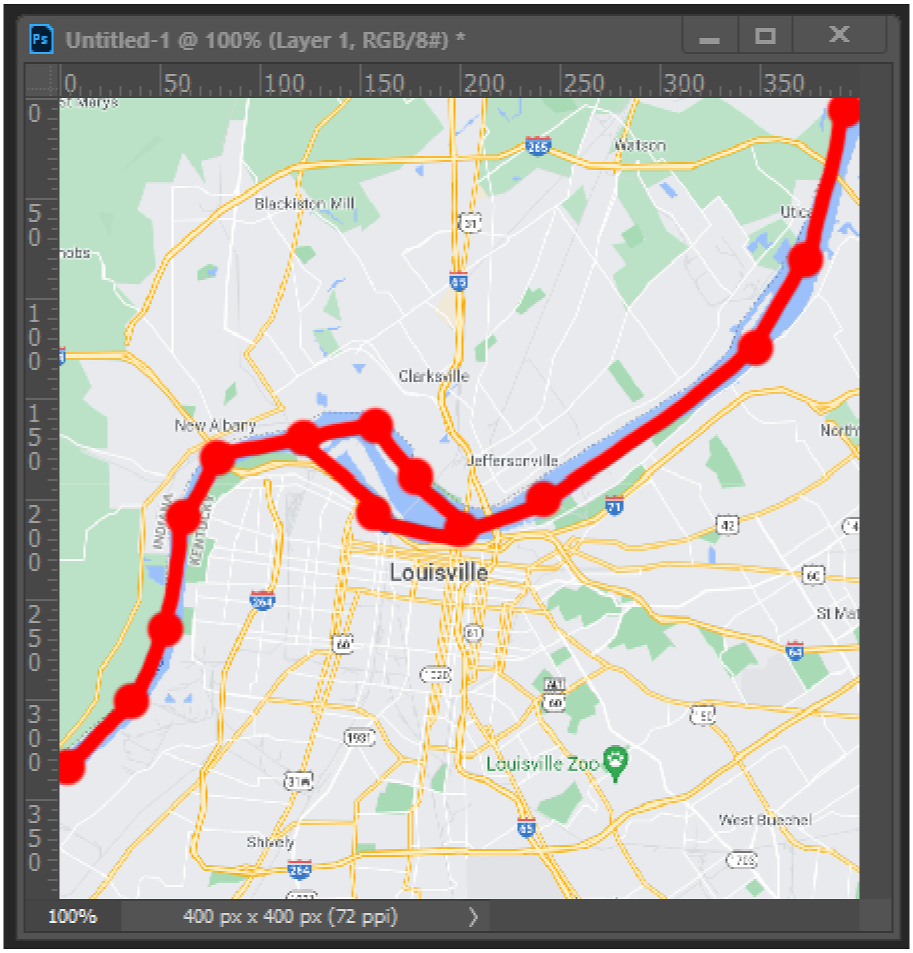
- Chicago: 300 x 400
- Cincinnati: 400 x 400
- Detroit: 400 x 400
- Pittsburgh: 400 x 300

Now let’s find a map of the city that we’re going to try to emulate. In this example we will mimic the S-curve that the Ohio River etches as it meanders around the city of Louisville:

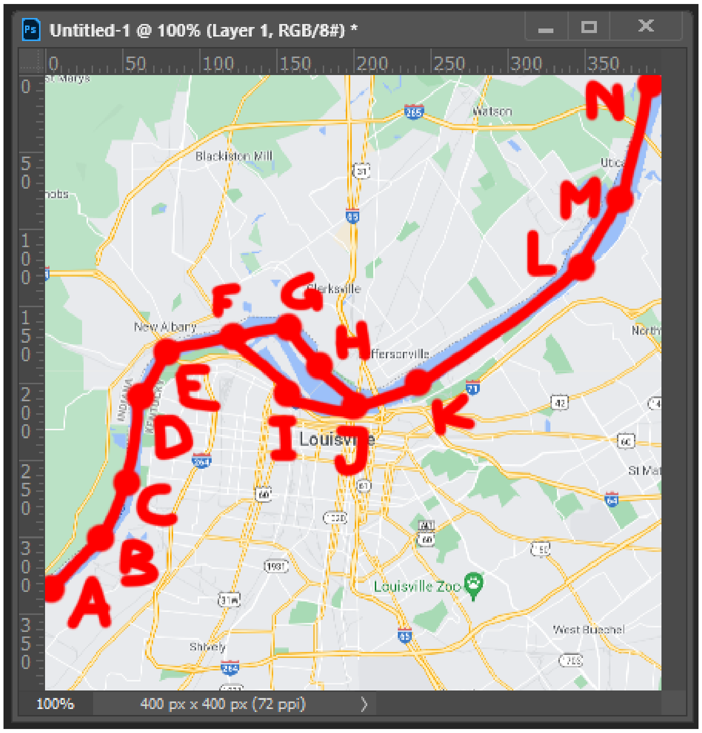


We'll need to find various points that correspond to segments of this river. It's probably easiest to open this map in an image editing program like Photoshop or Paint.NET and then:

- Make a new file and paste the map screenshot
- Resize the image to be the same as map size, e.g. **400x400** in this case
- Draw points and lines that roughly follow the shape of the river

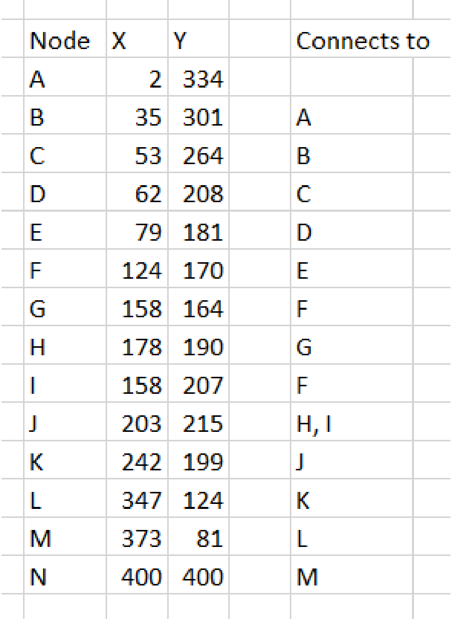


Now generate a list of coordinates for these nodes. We'll give each node a name and x-and-y-position, and list what previous node it connects to:



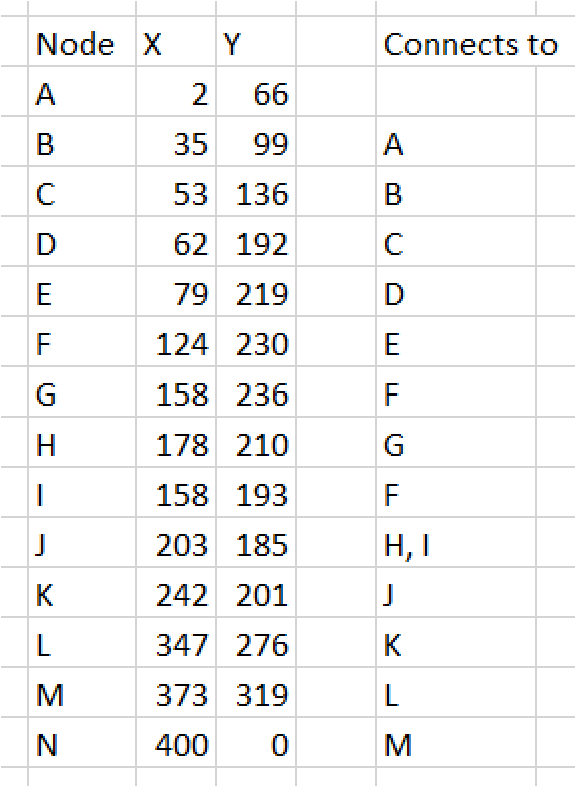
Untitled-1 @ 100% (Layer 1, RGB/8#) *

100% 400 px x 400 px (72 ppi)



Node	X	Y	Connects to
A	2	334	
B	35	301	A
C	53	264	B
D	62	208	C
E	79	181	D
F	124	170	E
G	158	164	F
H	178	190	G
I	158	207	F
J	203	215	H, I
K	242	199	J
L	347	124	K
M	373	81	L
N	400	400	M

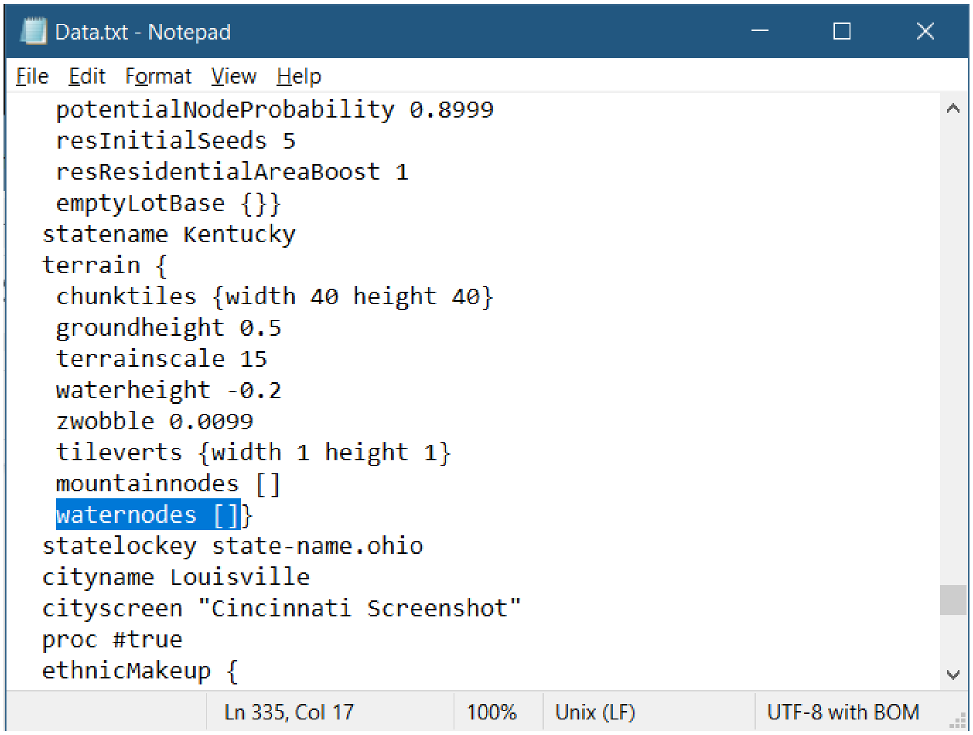
Now one more step: Photoshop has the (0, 0) coordinate in the upper left corner, but in-game maps use the (0, 0) coordinate in the lower left corner. So in this case we need to flip the Y coordinate, by computing **y = 400 - y** for each node:



Node	X	Y		Connects to
A	2	66		
B	35	99		A
C	53	136		B
D	62	192		C
E	79	219		D
F	124	230		E
G	158	236		F
H	178	210		G
I	158	193		F
J	203	185		H, I
K	242	201		J
L	347	276		K
M	373	319		L
N	400	0		M

Ok, we’re now ready to re-type this into the save file.

Open the **Data\Data.txt** file again, and look for section named “waternodes”. It should look something like this:



```
   potentialNodeProbability 0.8999
   resInitialSeeds 5
   resResidentialAreaBoost 1
   emptyLotBase {}}
  statename Kentucky
  terrain {
   chunktiles {width 40 height 40}
   groundheight 0.5
   terrainscale 15
   waterheight -0.2
   zwobble 0.0099
   tileverts {width 1 height 1}
   mountainnodes []
   waternodes []}
  statelockey state-name.ohio
  cityname Louisville
  cityscreen "Cincinnati Screenshot"
  proc #true
  ethnicMakeup {
```

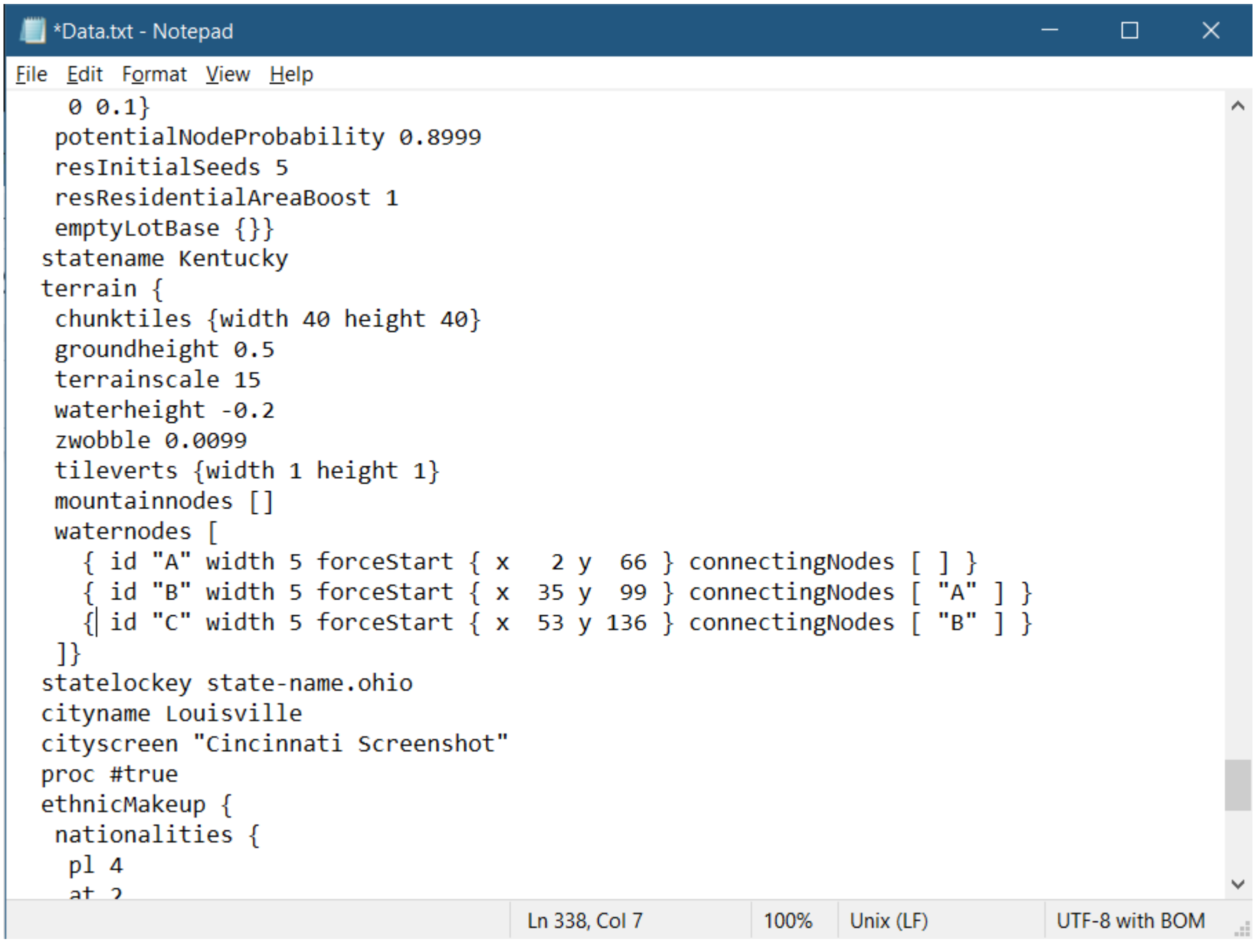
This means “waternodes” is an empty array. Let’s add some new entries in that section for the first three nodes, which we will call A B and C:

```
{ id "A" width 5 forceStart { x  2 y  66 } connectingNodes [ ] }
{ id "B" width 5 forceStart { x 35 y  99 } connectingNodes [ "A" ] }
{ id "C" width 5 forceStart { x 53 y 136 } connectingNodes [ "B" ] }
```

The fields for each entry are:

- **id** is the node name, in double quotes
- **width** is the width of the river at that point, in map units
- **forceStart** is the position of the node, in map units
- **connectingNodes** is the list of other water nodes it connects to (there may be multiple)

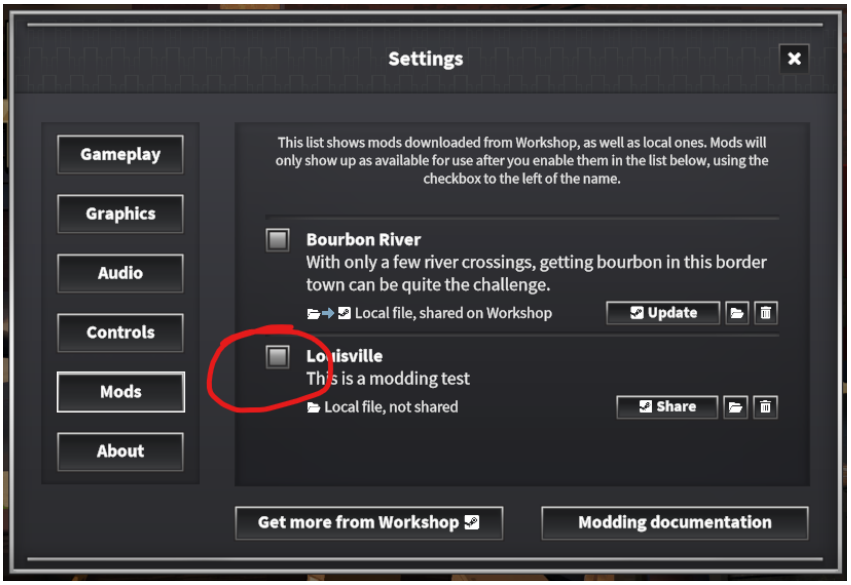
Your resulting list of nodes should look like this:



```
    0 0.1}
   potentialNodeProbability 0.8999
   resInitialSeeds 5
   resResidentialAreaBoost 1
   emptyLotBase {}}
  statename Kentucky
  terrain {
   chunktiles {width 40 height 40}
   groundheight 0.5
   terrainscale 15
   waterheight -0.2
   zwobble 0.0099
   tileverts {width 1 height 1}
   mountainnodes []
   waternodes [
     { id "A" width 5 forceStart { x   2 y  66 } connectingNodes [ ] }
     { id "B" width 5 forceStart { x  35 y  99 } connectingNodes [ "A" ] }
     { id "C" width 5 forceStart { x  53 y 136 } connectingNodes [ "B" ] }
   ]}
  statelockey state-name.ohio
  cityname Louisville
  cityscreen "Cincinnati Screenshot"
  proc #true
  ethnicMakeup {
   nationalities {
    pl 4
    at 2
```

Save this, and let’s load this up in the game and see the beginnings of the Ohio River!

First don’t forget to enable the new mod:



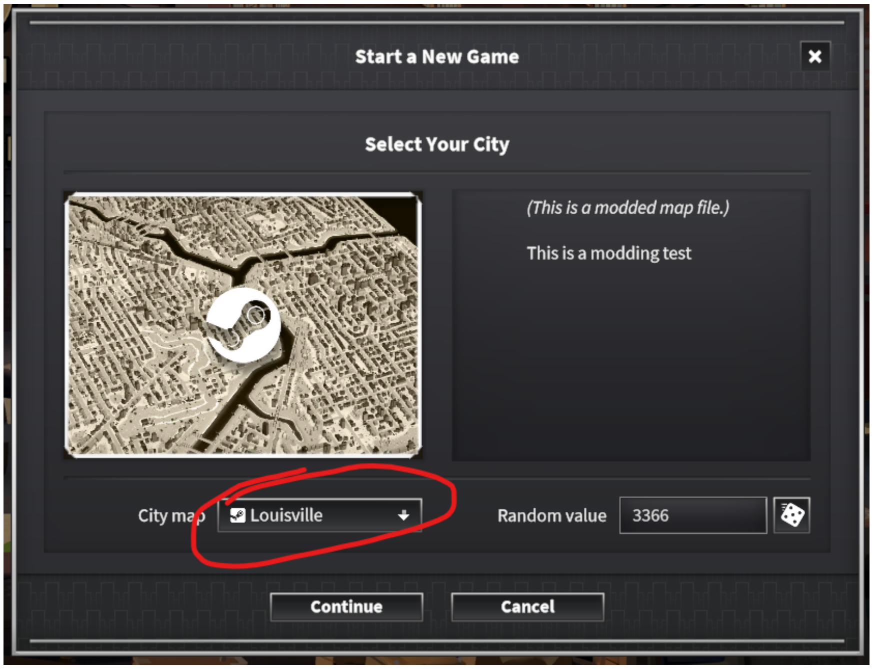
Settings

This list shows mods downloaded from Workshop, as well as local ones. Mods will only show up as available for use after you enable them in the list below, using the checkbox to the left of the name.

□ **Bourbon River**
With only a few river crossings, getting bourbon in this border town can be quite the challenge.
Local file, shared on Workshop

□ **Louisville**
This is a modding test
Local file, not shared

Then we start a new game:



Start a New Game

Select Your City

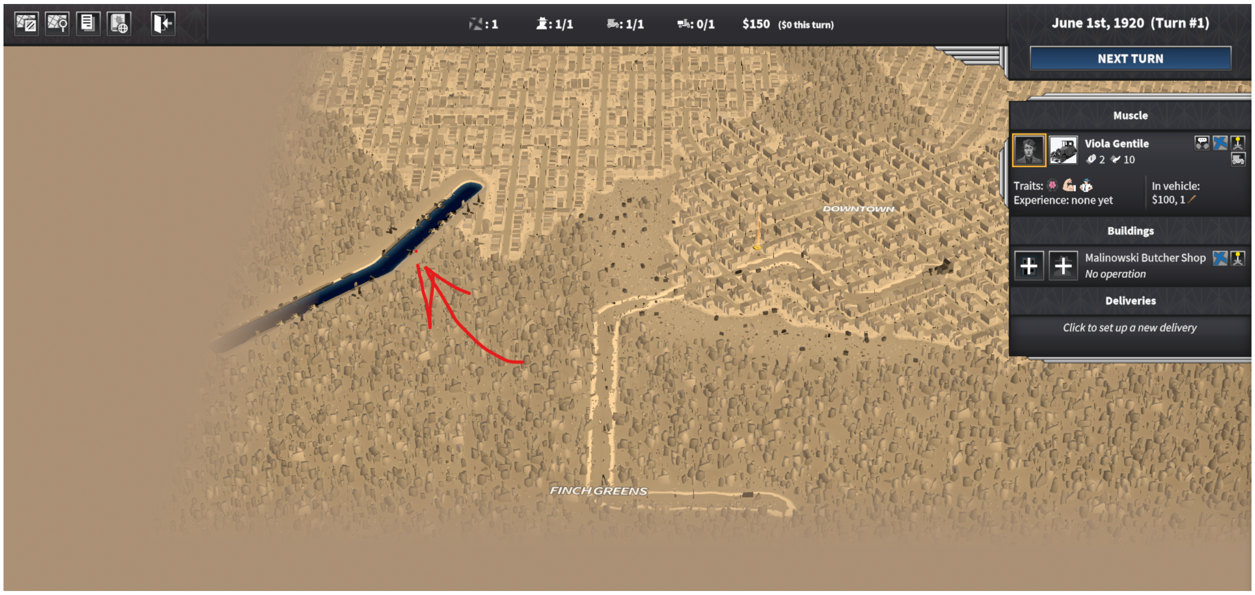
(This is a modded map file.)

This is a modding test

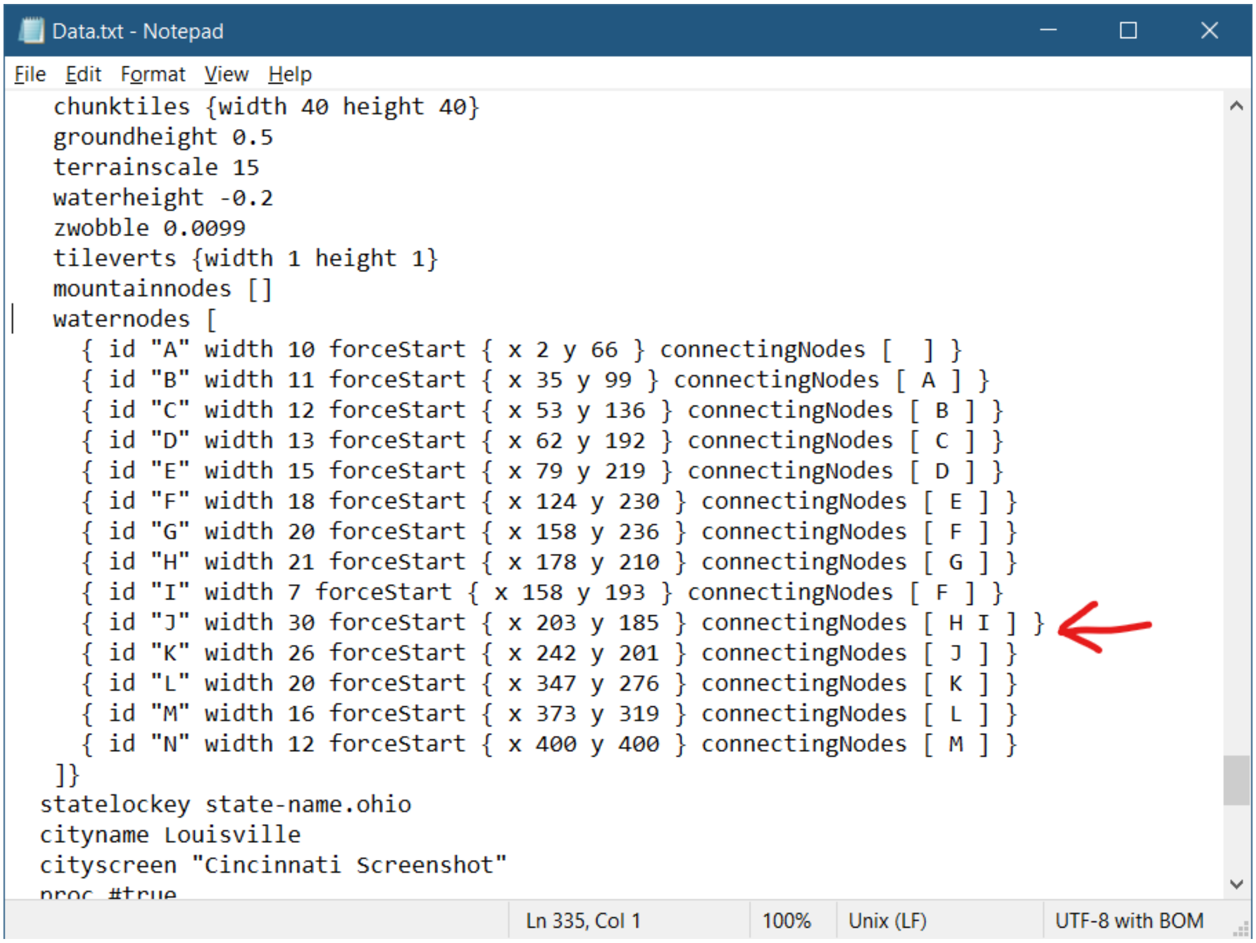
City map: Louisville

Random value: 3366

Now if we look in the corner of the map, we should see the beginnings of the river!



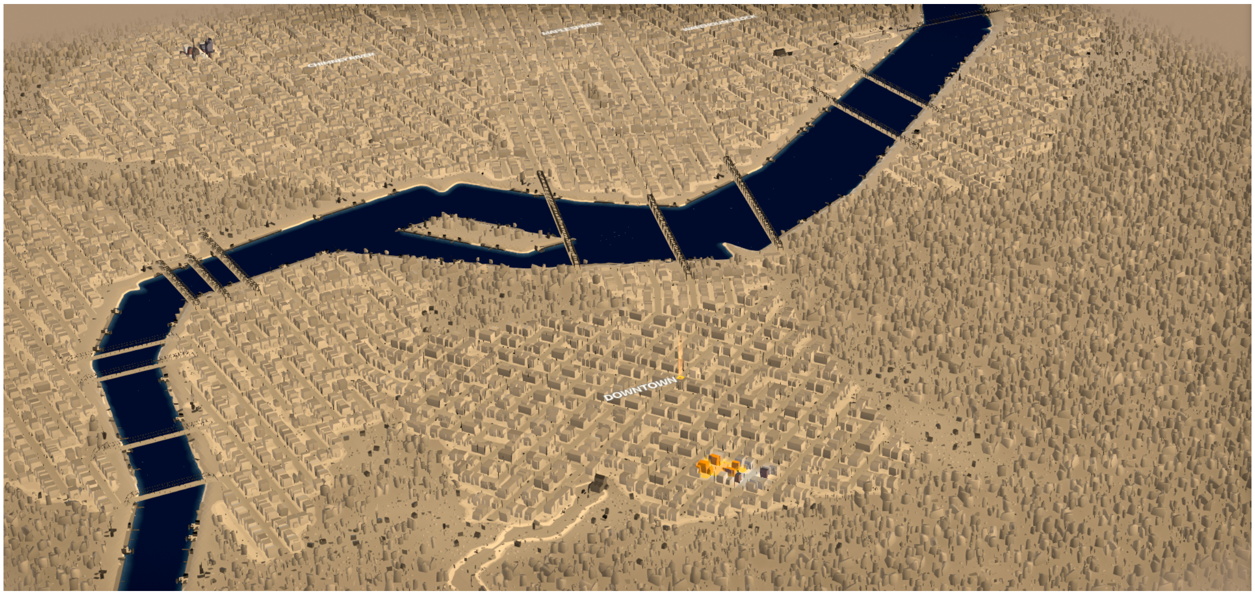
Go back to Data.txt and add all of the other remaining points. Also fix widths to be much more proportional:



```
   chunktiles {width 40 height 40}
   groundheight 0.5
   terrainscale 15
   waterheight -0.2
   zwobble 0.0099
   tileverts {width 1 height 1}
   mountainnodes []
   waternodes [
     { id "A" width 10 forceStart { x 2 y 66 } connectingNodes [   ] }
     { id "B" width 11 forceStart { x 35 y 99 } connectingNodes [ A ] }
     { id "C" width 12 forceStart { x 53 y 136 } connectingNodes [ B ] }
     { id "D" width 13 forceStart { x 62 y 192 } connectingNodes [ C ] }
     { id "E" width 15 forceStart { x 79 y 219 } connectingNodes [ D ] }
     { id "F" width 18 forceStart { x 124 y 230 } connectingNodes [ E ] }
     { id "G" width 20 forceStart { x 158 y 236 } connectingNodes [ F ] }
     { id "H" width 21 forceStart { x 178 y 210 } connectingNodes [ G ] }
     { id "I" width 7 forceStart { x 158 y 193 } connectingNodes [ F ] }
     { id "J" width 30 forceStart { x 203 y 185 } connectingNodes [ H I ] }
     { id "K" width 26 forceStart { x 242 y 201 } connectingNodes [ J ] }
     { id "L" width 20 forceStart { x 347 y 276 } connectingNodes [ K ] }
     { id "M" width 16 forceStart { x 373 y 319 } connectingNodes [ L ] }
     { id "N" width 12 forceStart { x 400 y 400 } connectingNodes [ M ] }
   ]}
  statelockey state-name.ohio
  cityname Louisville
  cityscreen "Cincinnati Screenshot"
  proc #true
```

NOTE: there is no comma between nodes H and I, or pretty much anywhere else in this file. This format is not the same as JSON, and commas and colons will cause errors.

Let’s load this in the game - now that’s more like it:

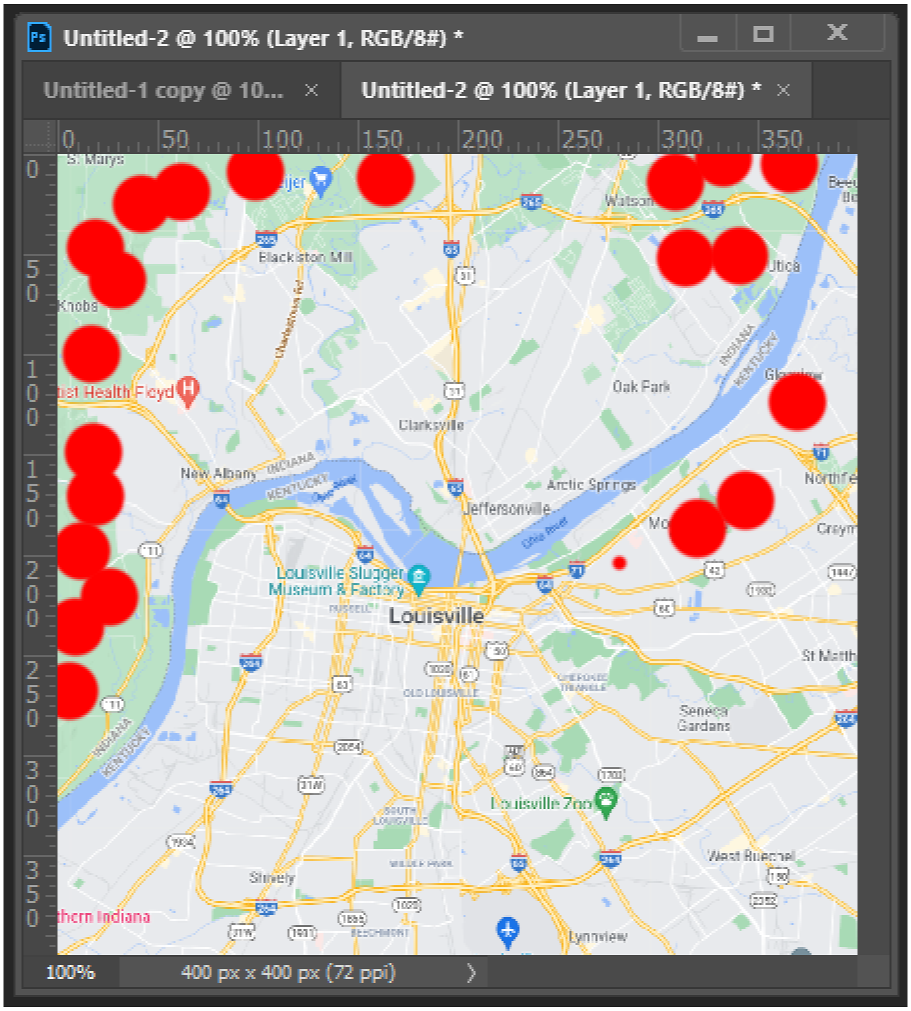


NOTE: as you edit mod files, MAKE SURE to balance the curly braces { } as well as square braces [ ], so that there is an equal number of opening and closing braces, and in the correct order. Unbalanced braces will cause errors.

Adding hills

Hills are collections of oblong blobs that get placed at specific locations.

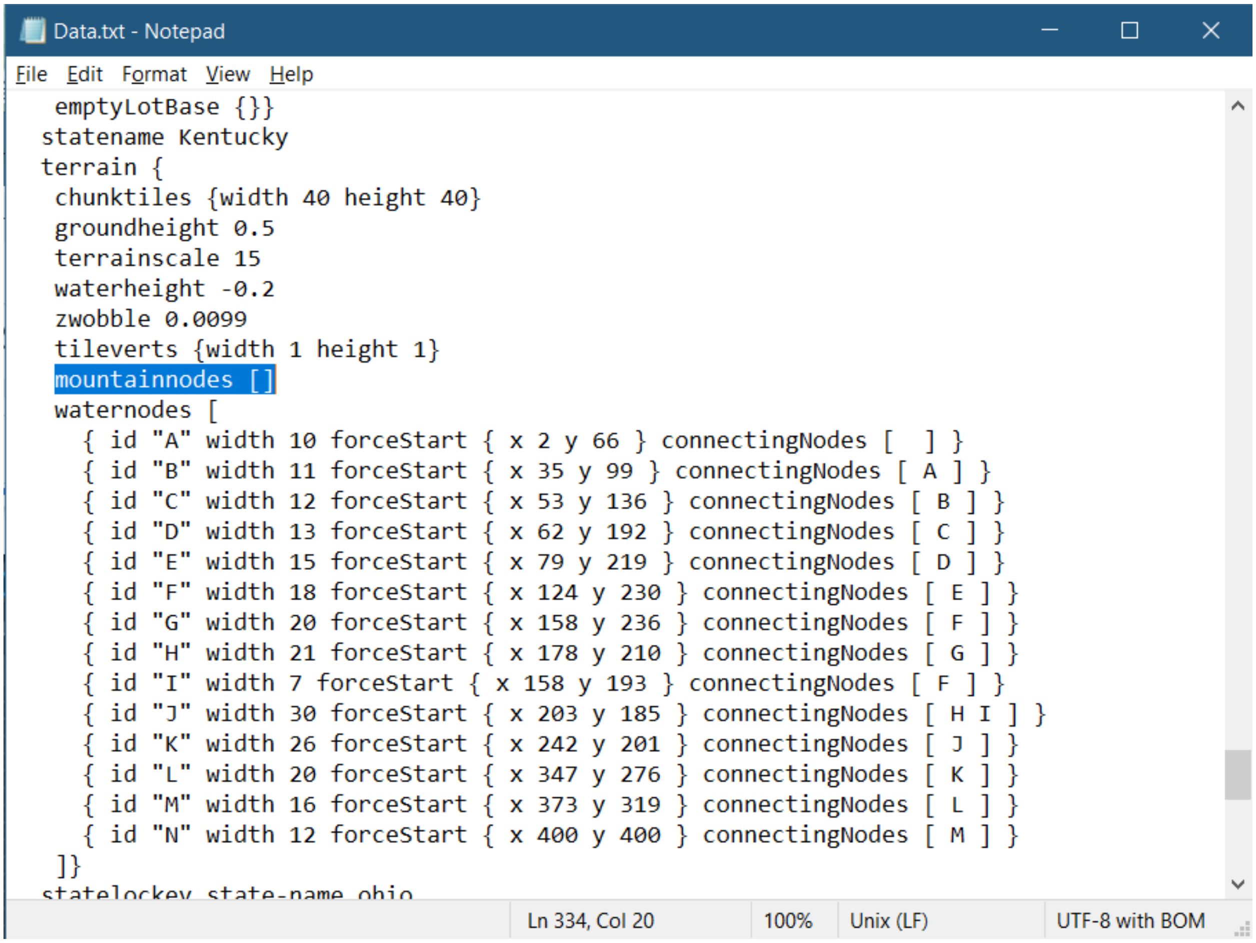
Let’s place some hills like so:



Similarly as before we’re going to define them as entries under the “mountainnodes” array, with the following fields:

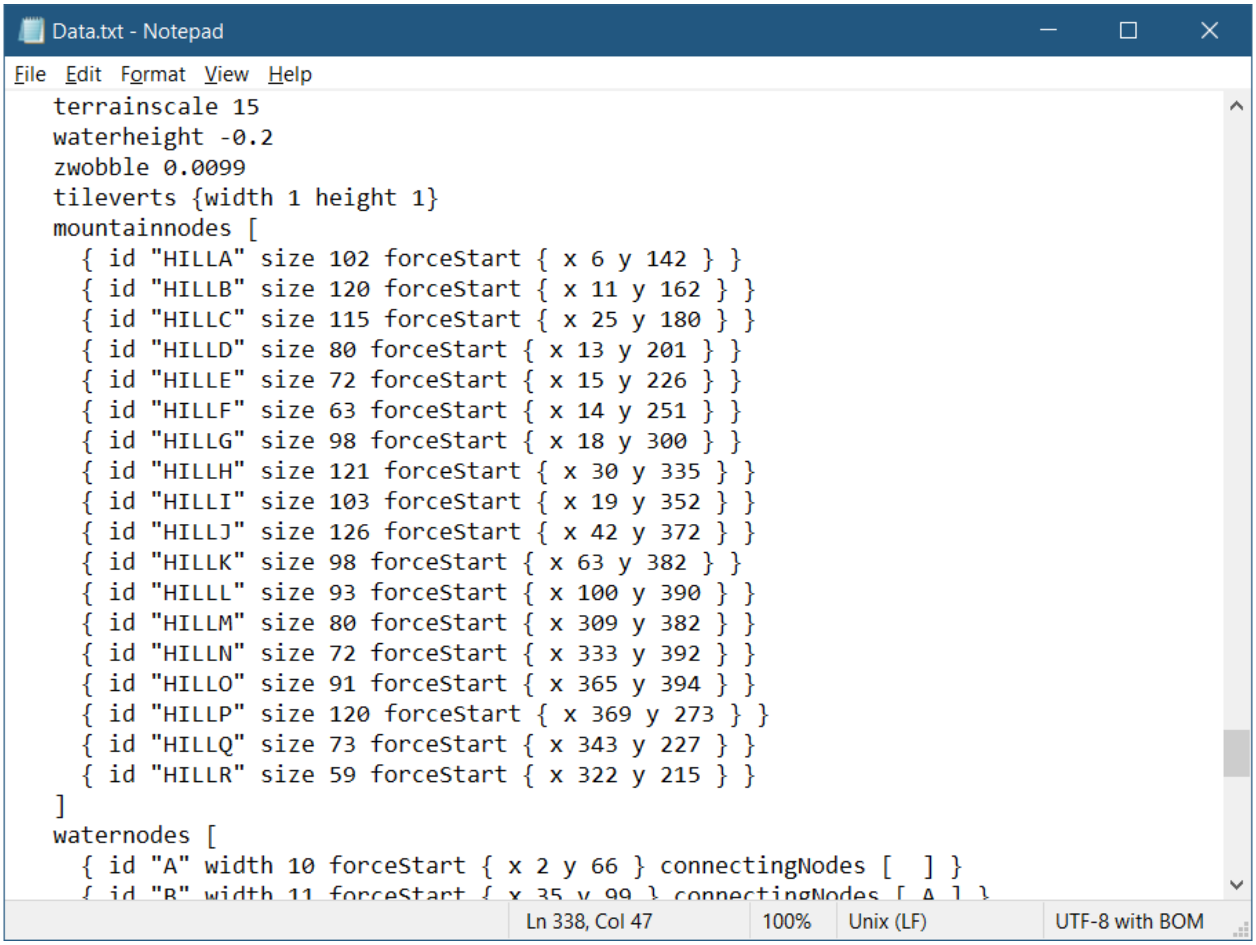
- **id** - unique ID as before
- **size** - expansion size of the hill (not exactly map units, takes a bit of experimentation)
- **forceStart** - hill center

Now replace the empty mountainnodes array:



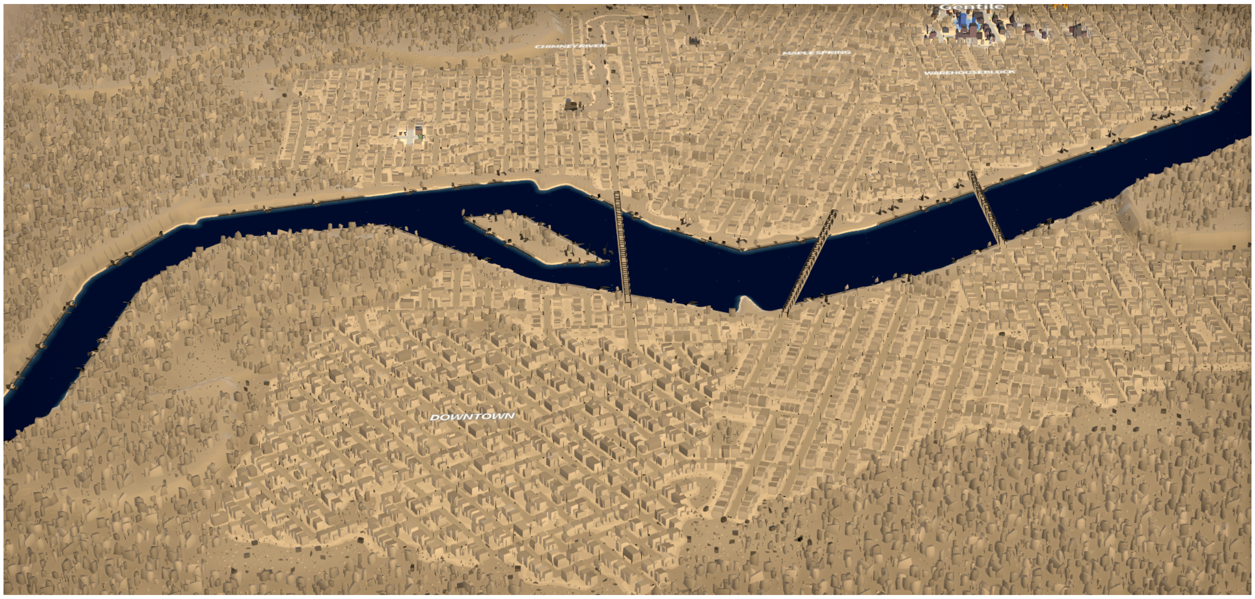
```
  emptyLotBase {}}
 statename Kentucky
 terrain {
  chunktiles {width 40 height 40}
  groundheight 0.5
  terrainscale 15
  waterheight -0.2
  zwobble 0.0099
  tileverts {width 1 height 1}
  mountainnodes []
  waternodes [
    { id "A" width 10 forceStart { x 2 y 66 } connectingNodes [   ] }
    { id "B" width 11 forceStart { x 35 y 99 } connectingNodes [ A ] }
    { id "C" width 12 forceStart { x 53 y 136 } connectingNodes [ B ] }
    { id "D" width 13 forceStart { x 62 y 192 } connectingNodes [ C ] }
    { id "E" width 15 forceStart { x 79 y 219 } connectingNodes [ D ] }
    { id "F" width 18 forceStart { x 124 y 230 } connectingNodes [ E ] }
    { id "G" width 20 forceStart { x 158 y 236 } connectingNodes [ F ] }
    { id "H" width 21 forceStart { x 178 y 210 } connectingNodes [ G ] }
    { id "I" width 7 forceStart { x 158 y 193 } connectingNodes [ F ] }
    { id "J" width 30 forceStart { x 203 y 185 } connectingNodes [ H I ] }
    { id "K" width 26 forceStart { x 242 y 201 } connectingNodes [ J ] }
    { id "L" width 20 forceStart { x 347 y 276 } connectingNodes [ K ] }
    { id "M" width 16 forceStart { x 373 y 319 } connectingNodes [ L ] }
    { id "N" width 12 forceStart { x 400 y 400 } connectingNodes [ M ] }
  ]}
 statelockey state-name ohio
```

Find that and replace with a new list of nodes, like this:



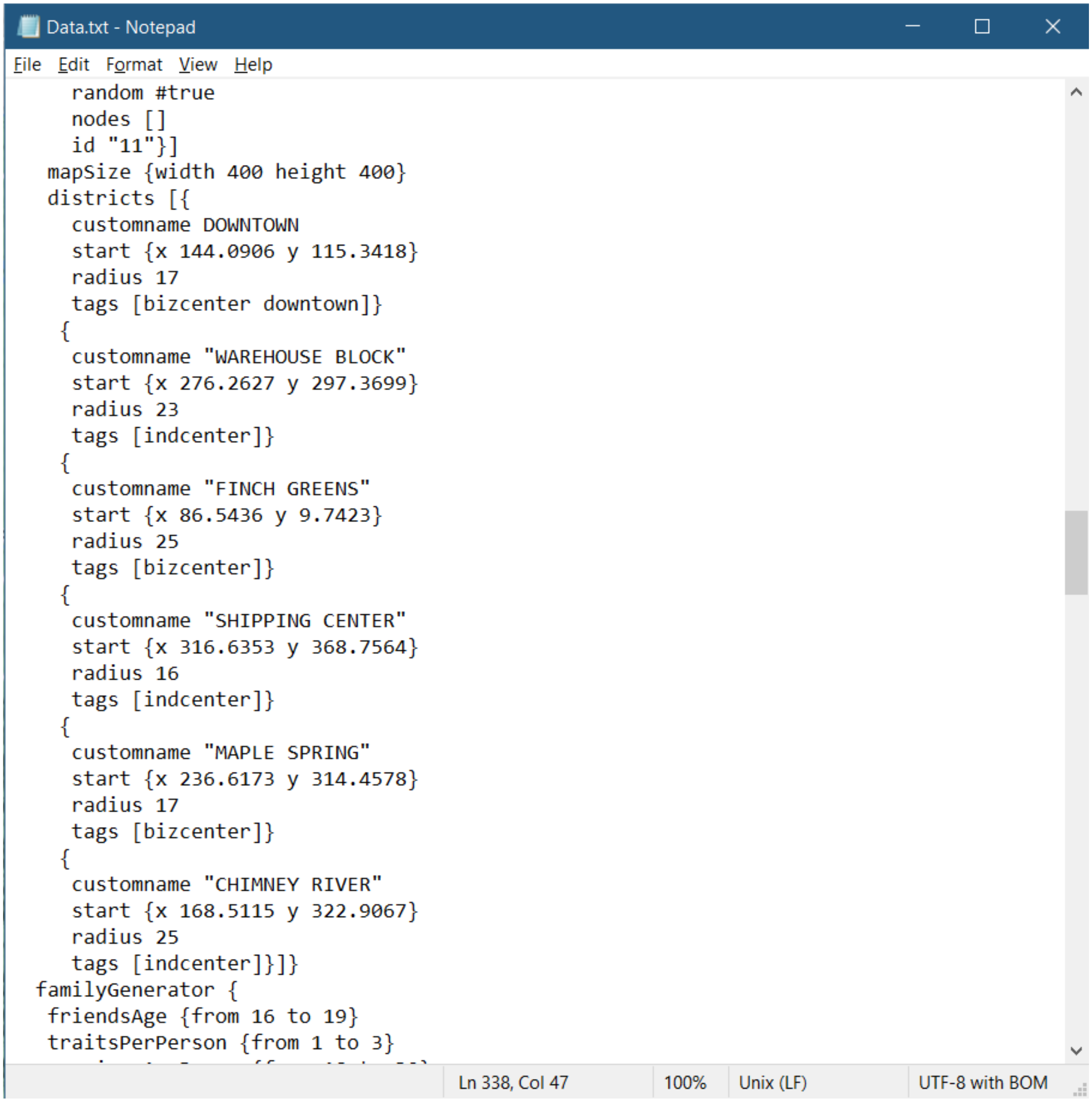
```
  terrainscale 15
  waterheight -0.2
  zwobble 0.0099
  tileverts {width 1 height 1}
  mountainnodes [
    { id "HILLA" size 102 forceStart { x 6 y 142 } }
    { id "HILLB" size 120 forceStart { x 11 y 162 } }
    { id "HILLC" size 115 forceStart { x 25 y 180 } }
    { id "HILLD" size 80 forceStart { x 13 y 201 } }
    { id "HILLE" size 72 forceStart { x 15 y 226 } }
    { id "HILLF" size 63 forceStart { x 14 y 251 } }
    { id "HILLG" size 98 forceStart { x 18 y 300 } }
    { id "HILLH" size 121 forceStart { x 30 y 335 } }
    { id "HILLI" size 103 forceStart { x 19 y 352 } }
    { id "HILLJ" size 126 forceStart { x 42 y 372 } }
    { id "HILLK" size 98 forceStart { x 63 y 382 } }
    { id "HILLL" size 93 forceStart { x 100 y 390 } }
    { id "HILLM" size 80 forceStart { x 309 y 382 } }
    { id "HILLN" size 72 forceStart { x 333 y 392 } }
    { id "HILLO" size 91 forceStart { x 365 y 394 } }
    { id "HILLP" size 120 forceStart { x 369 y 273 } }
    { id "HILLQ" size 73 forceStart { x 343 y 227 } }
    { id "HILLR" size 59 forceStart { x 322 y 215 } }
  ]
  waternodes [
    { id "A" width 10 forceStart { x 2 y 66 } connectingNodes [  ] }
    { id "B" width 11 forceStart { x 35 y 99 } connectingNodes [ A ] }
```

And there we go, we get some nice low hills on the outskirts of town:



Naming neighborhoods

For neighborhoods, look for the “districts” field in the data file. This field will already be pre-filled with some existing neighborhoods, and it might look something like this:

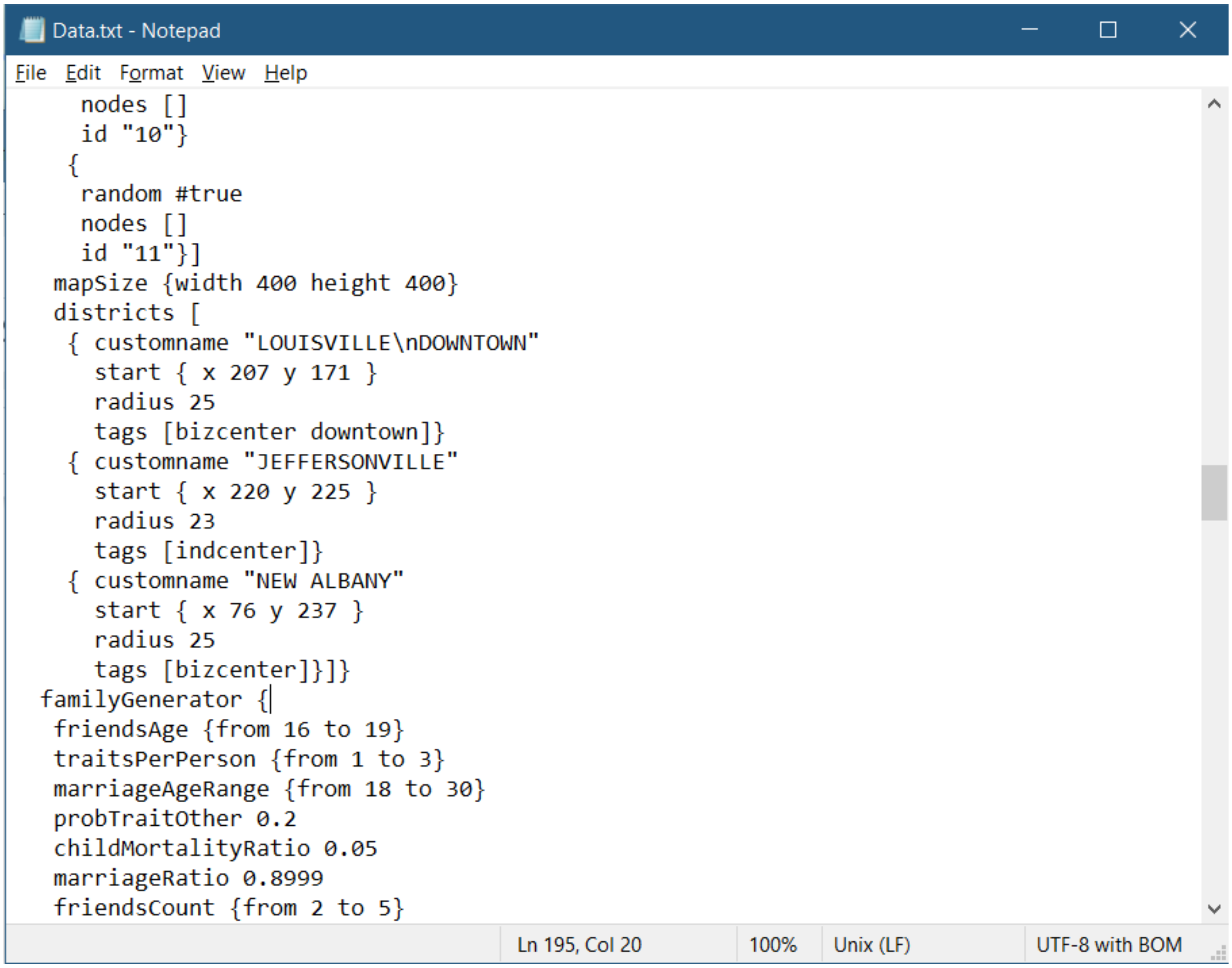


```
     random #true
     nodes []
     id "11"}]
   mapSize {width 400 height 400}
   districts [{
     customname DOWNTOWN
     start {x 144.0906 y 115.3418}
     radius 17
     tags [bizcenter downtown]}
    {
     customname "WAREHOUSE BLOCK"
     start {x 276.2627 y 297.3699}
     radius 23
     tags [indcenter]}
    {
     customname "FINCH GREENS"
     start {x 86.5436 y 9.7423}
     radius 25
     tags [bizcenter]}
    {
     customname "SHIPPING CENTER"
     start {x 316.6353 y 368.7564}
     radius 16
     tags [indcenter]}
    {
     customname "MAPLE SPRING"
     start {x 236.6173 y 314.4578}
     radius 17
     tags [bizcenter]}
    {
     customname "CHIMNEY RIVER"
     start {x 168.5115 y 322.9067}
     radius 25
     tags [indcenter]}]}
  familyGenerator {
   friendsAge {from 16 to 19}
   traitsPerPerson {from 1 to 3}
```

Each entry has the following fields:

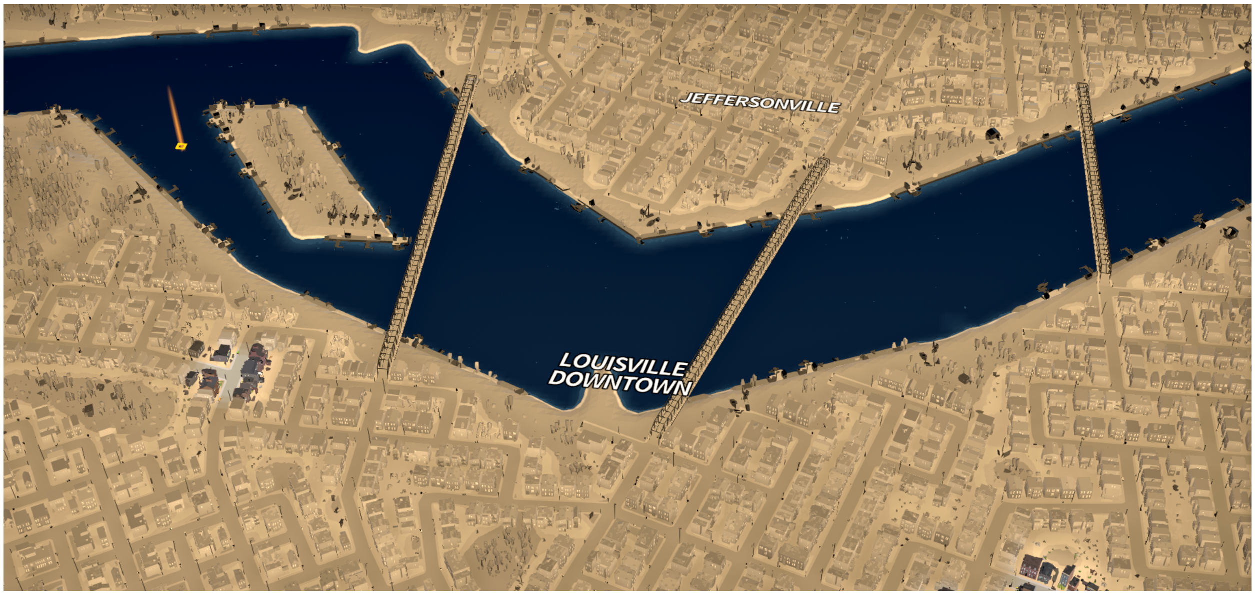
- **customname** is the name displayed on screen
    - You can use “\n” to force a line break, e.g. “HELLO\nWORLD”
- **start** is the x, y position of the district
- **radius** is the radius in map units
- **tags** is an array of tags, which can be one or more of the following:
    - **downtown** tag specifies the center of town
    - **bizcenter** tag specifies a commercial center, e.g. high-density hotels and stores
    - **indcenter** tag specifies an industrial center, e.g. large factories and warehouses

Let’s replace exiting definitions with something more thematic, at correct locations, maybe like this:



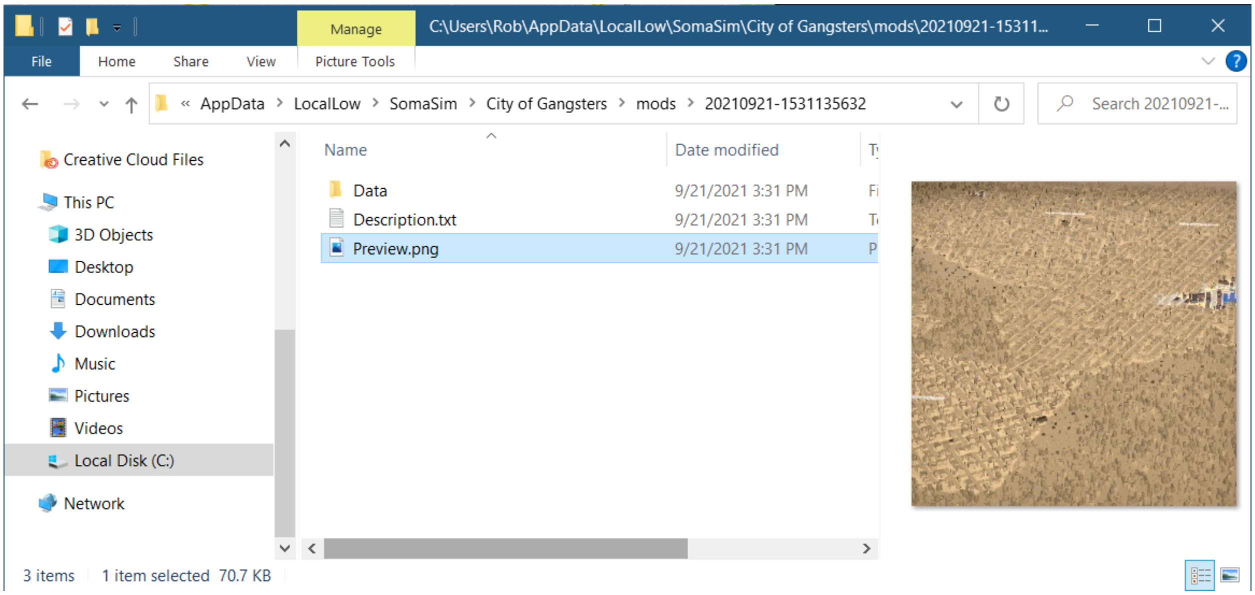
```
   nodes []
   id "10"}
  {
   random #true
   nodes []
   id "11"}]
 mapSize {width 400 height 400}
 districts [
  { customname "LOUISVILLE\nDOWNTOWN"
    start { x 207 y 171 }
    radius 25
    tags [bizcenter downtown]}
  { customname "JEFFERSONVILLE"
    start { x 220 y 225 }
    radius 23
    tags [indcenter]}
  { customname "NEW ALBANY"
    start { x 76 y 237 }
    radius 25
    tags [bizcenter]}]}
familyGenerator {
 friendsAge {from 16 to 19}
 traitsPerPerson {from 1 to 3}
 marriageAgeRange {from 18 to 30}
 probTraitOther 0.2
 childMortalityRatio 0.05
 marriageRatio 0.8999
 friendsCount {from 2 to 5}
```

And when we run the game, we should see the following:



Updating the screenshot

One more thing: we changed the mod, we should also update the preview screenshot! The **Preview.png** file in the mod folder is the old image of a flat map:



C:\Users\Rob\AppData\LocalLow\SomaSim\City of Gangsters\mods\20210921-15311...

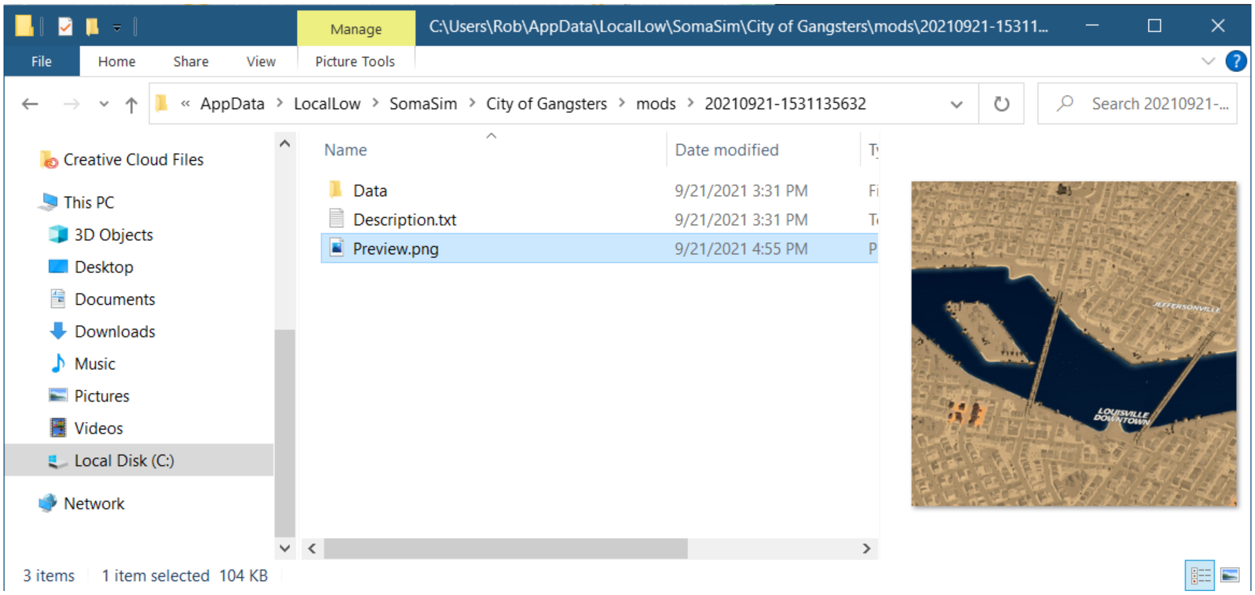
« AppData › LocalLow › SomaSim › City of Gangsters › mods › 20210921-1531135632

Search 20210921-...

Name	Date modified	T
Data	9/21/2021 3:31 PM	Fi
Description.txt	9/21/2021 3:31 PM	T
Preview.png	9/21/2021 3:31 PM	P

3 items 1 item selected 70.7 KB

Simply replace it with a better screenshot in Photoshop or another image editor, for example:



C:\Users\Rob\AppData\LocalLow\SomaSim\City of Gangsters\mods\20210921-15311...

« AppData › LocalLow › SomaSim › City of Gangsters › mods › 20210921-1531135632

Search 20210921-...

Name	Date modified	T
Data	9/21/2021 3:31 PM	Fi
Description.txt	9/21/2021 3:31 PM	T
Preview.png	9/21/2021 4:55 PM	P

3 items 1 item selected 104 KB

Uploading and playtesting

Now that we have a mod we're happy with, let's export it to Steam and test it.

Follow instructions from Part 1 to go through the following steps:

- In the game, in Settings > Mods, share the local Louisville map on Workshop
- In Workshop, find the new item and then
    - Change its visibility to "Public"
    - Subscribe to the item yourself
- Quit and restart the game, and you should now see two Louisvilles, one local and one remote
    - Disable the local one and enable the remote one
    - Start a new game in Louisville and verify it's working correctly!

Also, let's try to update this existing mod with some new changes:

- Go back to the mod text files, and try changing some aspects of it, such as name or description
- Quit and restart the game, to force a refresh, and again go into Settings > Mods
- Select your local version of the mod (it should reflect the changes), and hit the "Update" button
- In Workshop, you should see that the mod got updated

Congratulations! You have now made a customized map, and shared it with the world :)