City Game Design Document

By Cam Abreu, September, 2016

**Description:** City will be a fan based clone of the super Nintendo game Sim City (1991). The project goal is to have fun learning how to program systems and discover needed algorithms for the games systems. The game will be made using game maker language.

|  |  |  |
| --- | --- | --- |
| **Buildings Overview** | | |
| **Structure Name** | **Size** (in Square Units, which are 16x16 pixels) | **Description** |
| Trees | 1x1 | Reduce Pollution in an area around them, change sprite according to season. Trees are generated on maps during map creation randomly. |
| Road | 1x1 | Required for Zone growth |
| Power Line | 1x1 | Required for Zone growth |
| Zones (residential, commercial, industrial) | 3x3 | See “Supply and Demand” chart below. Zones have a 4 in 5 chance to update every cycle (for random effect). |
| Police Station | 3x3 | Reduces crime in an area around them |
| Fire Station | 3x3 | Reduces chance of fires in an area around them |
| School | 3x3 | Increases land value for residential areas |
| Power Plants (coal, nuclear) | 4x4 | Have a “power capacity” which other buildings require and drain |
| Hospital | 4x4 | Increases land value for all areas |
| Airport | 5x5 | Increases rate in which residential zones develop |

|  |  |
| --- | --- |
| **Gameplay Systems** | |
| **Gameplay System** | **Description** |
| Residential Demand | See “Supply and Demand” chart below |
| Industrial Demand | See “Supply and Demand” chart below |
| Commercial Demand | See “Supply and Demand” chart below |
| Money | Earned from taxes |
| Population | Village, Town, City, Capital, Metropolis, Megalopolis |
| Taxes | Begins with overall, unlock ability to adjust residential, commercial, and industrial individually. |
| Date | Month, Year |
| Season | Four seasons total |
| Cycle | Each season is broken into 24 cycles. Every 8 cycles are a month. Cycles are 3 seconds long on normal speed. (60 frames/second, so 180 frames). That means that every month on normal speed is 24 seconds, and every season is 72 seconds, and every year is 288 seconds or about 5 minutes. |
| Game Speed | Increases/decreases the speed in which cycles occur |
| Pollution |  |
| Crime |  |
| Power Capacity | Held within the power plant building. Things connected to the power plant through power lines “consume” power from that capacity. Denser zones consume more power. |
| Land value | Calculated from pollution, crime, fire chance, distance from developed commercial zone (shopping), distance from developed industrial zone (jobs) |
| Citizen Approval | Calculated from taxes, overall pollution, growth from last fiscal year, police coverage, land value in residential zones, school coverage |

|  |  |
| --- | --- |
| **Other Systems** | |
| **System** | **Description** |
| Save Game |  |
| Load Game |  |
| Map Generation |  |
| Approval Calculator |  |
| Land Value Calculator |  |
| Camera | WASD controls |
| Data menu | Displays information about the city |
| Purchase Bar |  |
| Background Music |  |
| SFX | When placing buildings |
| Message Feed | Gives little hints about what to do, such as “Build residential Zones…” |
| Information Bar | Displays: Date, Population, Money, Speed, Supply and Demand |
| Grid system | For placing buildings |
| Mouse Over Descriptions | For buildings on the purchase bar |
|  |  |
|  |  |

**Encapsulation Design**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Game Maker Objects** | | | | |
| **Object Name** | **Parent** | **Sprite** | **Created by:** | **Description** |
| obj\_main | none | none | In room at start | * Variables:   + mouseId * ds\_SUList = ds\_list\_create(); |
| obj\_SU | none | spr\_SU | generated by the scr\_mapGenerate script. | * 16x16 pixels. Each SU (square unit) holds its own score for   + pollution (using distance score to all trees)   + distance to all developed residential zones   + distance to all developed commercial zones   + distance to all developed industrial zones   + distance to any police stations   + distance to any fire stations   + distance to any schools   + fire chance per cycle   + distance to water   + land value (averaged from above factors) * Updates every cycle * not created on top of water objects * Id’s are stored in main as “ds\_SUList” |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Object Name** | **Parent** | **Sprite** | **Created by:** | **Description** |
| obj\_IDList | none | none | main | Creates a list data structure for each type of building, and updates when buildings are created and destroyed.   * ds\_SUID = ds\_list\_create(); * ds\_allBuildingsID = ds\_list\_create(); * ds\_treesID = ds\_list\_create(); * ds\_FireSID = ds\_list\_create(); * ds\_polSID = ds\_list\_create(); * ds\_schoolID = ds\_list\_create(); * ds\_hospID = ds\_list\_create(); * ds\_airportID = ds\_list\_create(); * ds\_resZoneID = ds\_list\_create(); * ds\_comZoneID = ds\_list\_create(); * ds\_indZoneID = ds\_list\_create(); * ds\_roadID = ds\_list\_create(); * ds\_powerLineID = ds\_list\_create(); * ds\_waterID = ds\_list\_create(); |
| obj\_water | none | spr\_water | generated by the scr\_mapGenerate script. | * Buildings cannot be built on water * when created by the scr\_mapGenerate script, each water object’s id is stored in the ds\_waterID list. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ZONES** | | | | |
| **Object Name** | **Parent** | **Sprite** | **Created by:** | **Description** |
| obj\_zone | obj\_building | spr\_zone (changes for which type of zone) | no | 3x3SU in size, creates obj\_dev objects inside of it. |
| obj\_dev | none | none | no |  |
| obj\_devRes | obj\_dev | spr\_devRes, created on top of obj\_zone, index depends on tier level | no | Has tier levels based on land value. |
| obj\_devCom | obj\_dev | spr\_devCom, created on top of obj\_zone, index depends on tier level | no | Has tier levels based on land value. |
| obj\_devInd | obj\_dev | spr\_devInd, created on top of obj\_zone, index depends on tier level | no | Has tier levels based on land value. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Buildings** | | | | |
| **Object Name** | **Parent** | **Sprite** | **Created by:** | **Description** |
| obj\_building | none | none |  | This is the parent for all buildings. Cannot place buildings ontop of objects that are children of obj\_building |
| obj\_tree | obj\_building | spr\_tree | no | reduces pollution score for SU objects depending on distance |
| obj\_polStation | obj\_building | spr\_polStation | no |  |
| obj\_fireStation | obj\_building | spr\_fireStation | no |  |
| obj\_school | obj\_building | spr\_school | no |  |
| obj\_powerPlant | obj\_building | none | no |  |
| obj\_coalPlant | obj\_powerPlant | spr\_coalPlant | no |  |
| obj\_nucPlant | obj\_powerPlant | spr\_nucPlant | no |  |
| obj\_hospital | obj\_building | spr\_hospital | no |  |
| obj\_airport | obj\_building | spr\_airport | no |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Menu/Bars** | | | | |
| **Object Name** | **Parent** | **Sprite** | **Created by:** | **Description** |
| obj\_purchaseBar | none | none, but includes draw event | main |  |
| obj\_purchaseIcon | none | sprite depends on the purchaseID variable | obj\_purchaseBar | * purchaseID variable determines which building type * purchaseCost * purchaseDesc: the string description used for mouseover |
| obj\_topBar | none | none, but includes draw event | main |  |
| obj\_topBarIcon | none | sprite depends on the topBarID variable. topBarID uses enums. | obj\_topBar | * topBarID varbale determines which button type   + save   + load   + speed: slow   + speed: normal   + speed: fast   + mapInfo |
| obj\_mapWindow | none | none, but includes draw event | obj\_topBarIcon (with the MapWindow type) | Draws a menu on top of the game screen  Clickable features:   * Pollution * Crime * Land value * Zones |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Other/Misc** | | | | |
| **Object Name** | **Parent** | **Sprite** | **Created by:** | **Description** |
| obj\_mouse | none | obj\_mouse (depends on mouse state. Uses enums) | obj\_main | The cursor will change depending on the task. When a building is selected for placement, the mouse will change to the appropriate size box to show the outline of what will be placed. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Scripts** | | | |
| **Script Name** | **Used by** | **Arguments** | **Functionality** |
| scr\_mapGenerate | obj\_main | mapSize (enum: mapSmall, mapMedium, mapLarge) | used to procedurally generate the map. Uses scr\_generateSU, scr\_generateTrees, and scr\_generateWater |
| scr\_ResDemand | obj\_main |  |  |
| scr\_ComDemand | obj\_main |  |  |
| scr\_IndDemand | obj\_main |  |  |
| scr\_CalcIncome | obj\_main |  | uses tax, and other factors to calculate the fiscal income |
| scr\_CalcApproval | obj\_main |  | Calculated from taxes, overall pollution, population growth from last fiscal year, zone growth from last year, police coverage, land value in residential zones, school coverage |
| scr\_CalcZoneGrowth | scr\_CalcApproval |  |  |
| scr\_generateSU | scr\_mapGenerate | mapSize | nested for loop to create SU objects in a grid across the map so that every 16x16 pixels contains one SU object. |
| scr\_generateWater | scr\_mapGenerate | mapSize | generates bodies of water across the map |
| scr\_generateTrees | scr\_mapGenerate | mapSize | generates bodies of trees. Cannot create on water |
| scr\_distTree | obj\_SU |  | calculates a score based on the distance of all trees, and stores it as “treeScore” |
| scr\_distInd | obj\_SU |  | calculates a score based on the distance of all developed industrial zones (a check is made to see if the zone is developed). Stored in “indScore” |
| scr\_distRes | obj\_SU |  | calculates a score based on the distance of all developed residential zones (a check is made to see if the zone is developed). Stored in “resScore” |
| scr\_distCom | obj\_SU |  | calculates a score based on the distance of all developed commercial zones (a check is made to see if the zone is developed). Stored in “comScore” |
| scr\_distPol | obj\_SU |  | calculates a score based on the distance of all police stations. Stored in “polScore” |
| scr\_distFire | obj\_SU |  | calculates a score based on the distance of all police stations. Stored in “fireScore” |
| scr\_distSchool | obj\_SU |  | calculates a score based on the distance of all schools. Stored in “schoolScore” |
| scr\_mapInfoPollution | obj\_mapInfo |  | draws the pollution information to the map object using the |
|  |  |  |  |
|  |  |  |  |

Supply and Demand Chart

Dependencies:

High

Med

High

Medium

Low