

CMPPM 120

Tiles

Objectives

By the end of today you should be able to...

1. Reviewing some common patterns in Snowy States
 - a. You should be able to describe **how to fix** the three problems
2. Tiles
 - a. Describe how **tiles** are used in games
 - b. Practice using Tiled, a **tile editor**
 - c. Demonstrate how to put tilemaps **into Phaser**
3. Your Endless Runner
 - a. Have answers to your questions about your endless runner

Git tools

<https://www.sourcetreeapp.com/>

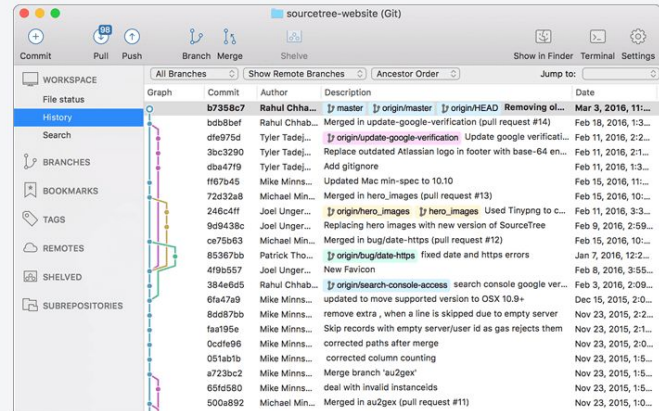
<https://www.gitkraken.com/>

<https://git-scm.com/downloads/guis>

Simplicity and power in a beautiful Git GUI

[Download for Windows](#)

Also available for Mac OS X



A free Git client for Windows and Mac

Sourcetree simplifies how you interact with your Git repositories so you can focus on coding. Visualize and manage your repositories through Sourcetree's simple Git GUI.



Simple for beginners

Say goodbye to the command line - simplify distributed version control with a Git client and quickly bring everyone up to speed.

Powerful for experts

Snowy States Pitfalls

Passing Local Variables

Try to avoid global variables

```
game.state.start('Play',  
true, false, this.score)
```

```
this.score = 0;  
game.state.start('Play', true, false, this.score);
```

Snowflakes cleanly wrapping

No popping at the edges of the
screen!

Star Catch Game
Use Arrow Keys to Move
Press [Space] to Start

Snowflakes cleanly wrapping

No popping at the edges of the
screen!

```
Snowstorm.prototype.update = function(){  
  //override the inherited update() method to add our own behaviors  
  if(this.reverseKey.justPressed()){  
    this.body.velocity.x = this.body.velocity.x * -1;  
  }  
  if(this.x >= game.width + 8 && this.body.velocity.x > 0){  
    this.x = -8;  
  }  
  if(this.y >= game.height + 8){  
    this.y = -8;  
  }  
  if(this.x <= -8 && this.body.velocity.x < 0){  
    this.x = game.width + 8;  
  }  
}
```

OR

```
//wrap  
game.world.wrap(this, 50, true);
```

Reverse toggles too quickly

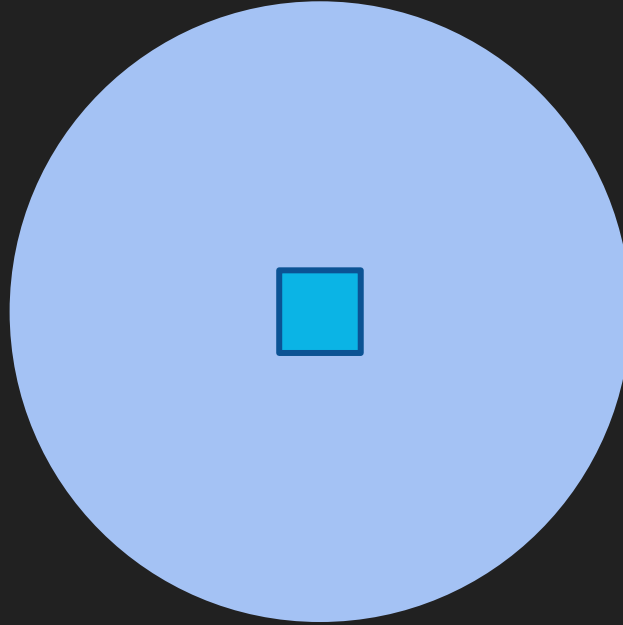
This kind of **threshold
problem** is one that you'll
encounter a lot

```
// reverse horizontal by pressing the 'R' key  
if(game.input.keyboard.isDown(Phaser.Keyboard.R)) {  
    this.body.velocity.x = -this.body.velocity.x;  
}
```

problem

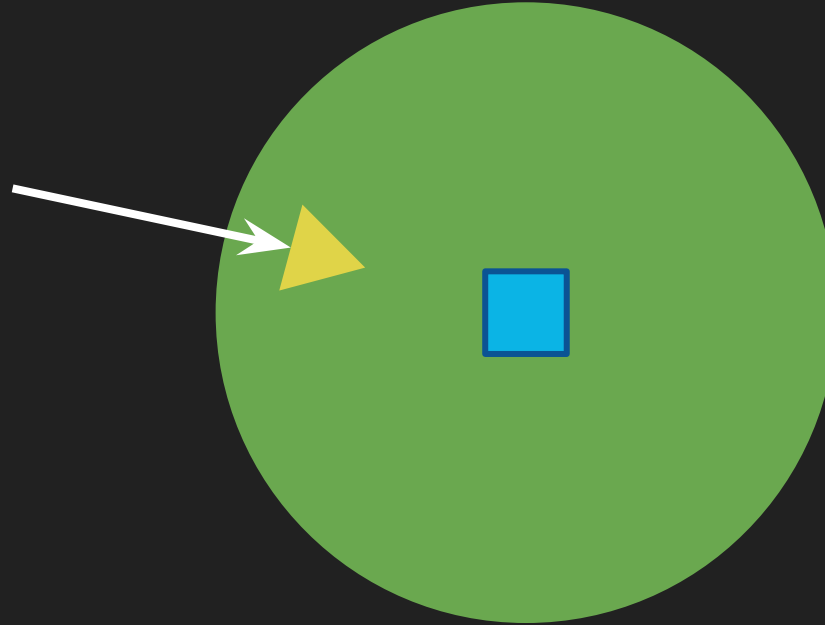


Threshold Problem



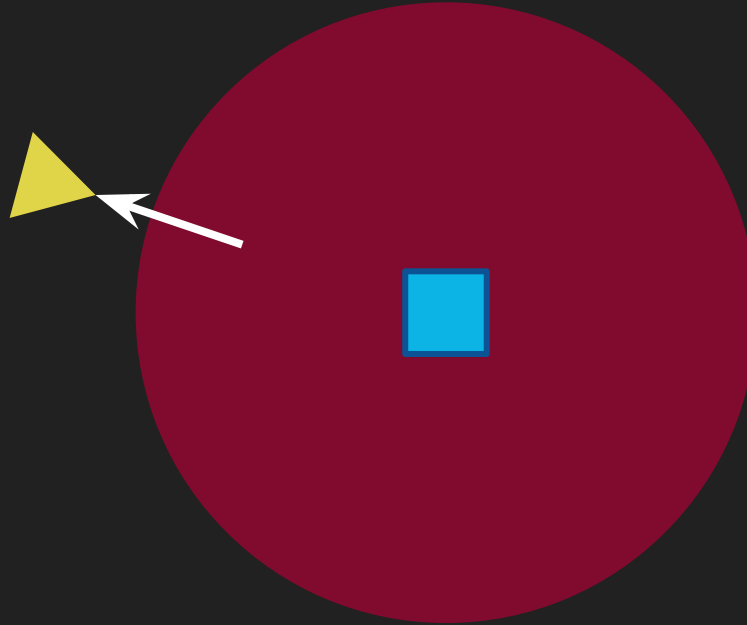
Threshold Problem

`enter()`

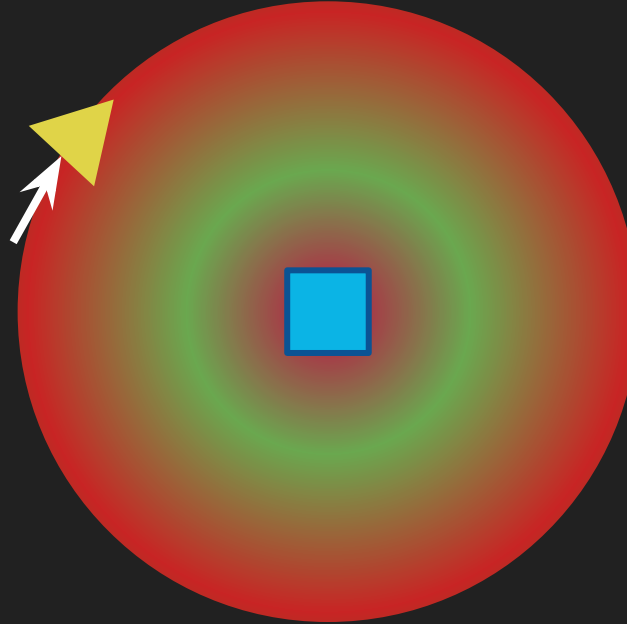


Threshold Problem

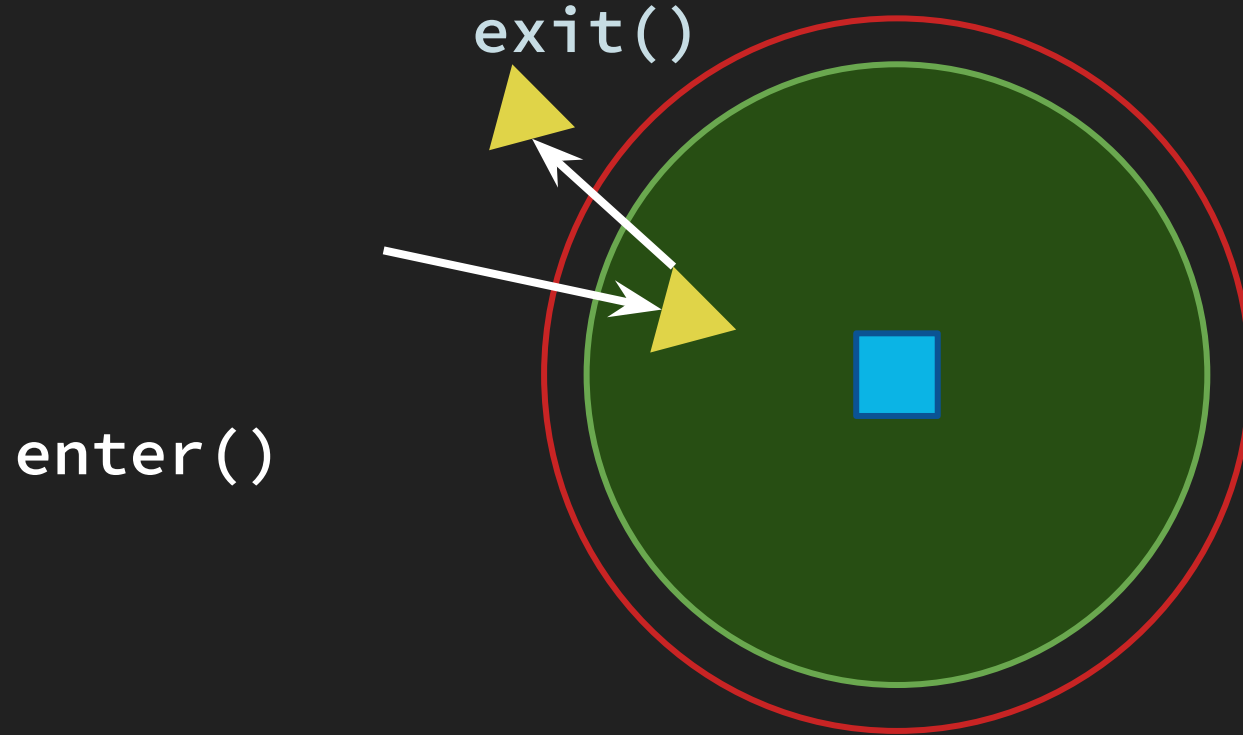
`exit()`



Race condition!



Add a threshold



Reverse toggles too quickly

This kind of **threshold
problem** is one that you'll
encounter a lot

```
// reverse horizontal by pressing the 'R' key
if(game.input.keyboard.isDown(Phaser.Keyboard.R)) {
    this.body.velocity.x = -this.body.velocity.x;
}
```

problem



```
if(this.reverseKey.justPressed() ){
    this.body.velocity.x = this.body.velocity.x * -1;
}
```

or

```
// reverses the horizontal velocity
if (game.input.keyboard.isDown(Phaser.KeyCode.R) && this.canPressR) {
    this.xVelocity = -this.xVelocity;
    this.canPressR = false;
}
if (!game.input.keyboard.isDown(Phaser.KeyCode.R) && !this.canPressR) {
    this.canPressR = true;
}
```

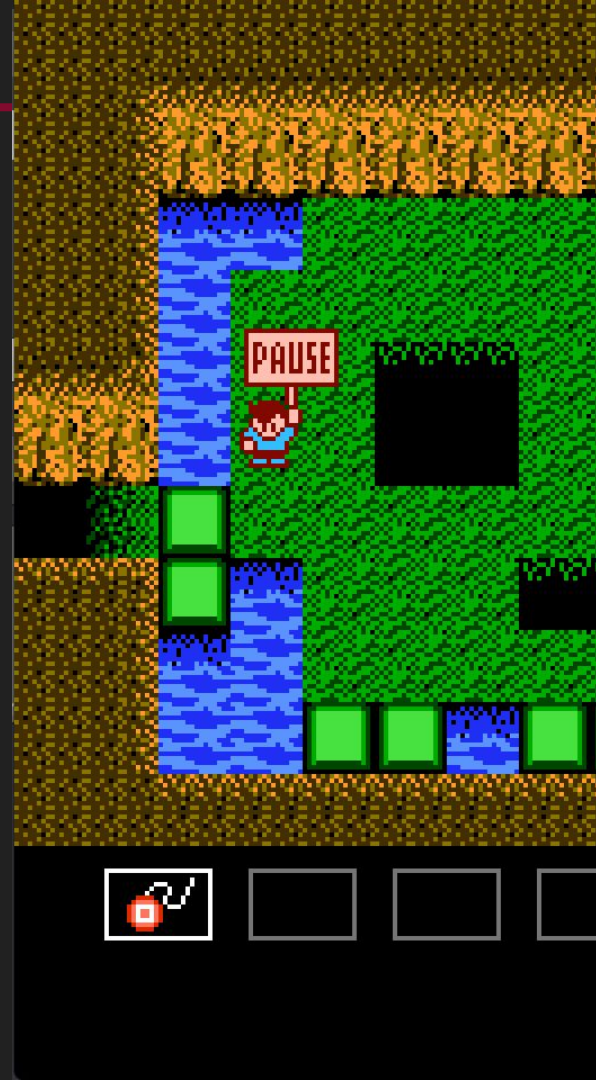
Tiles

C

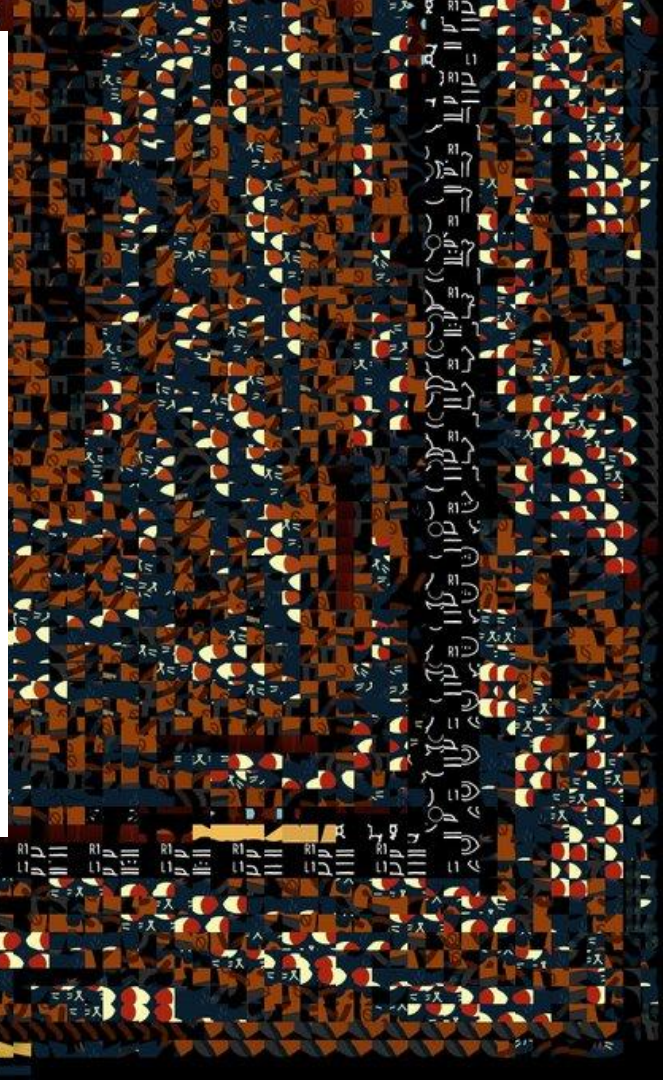
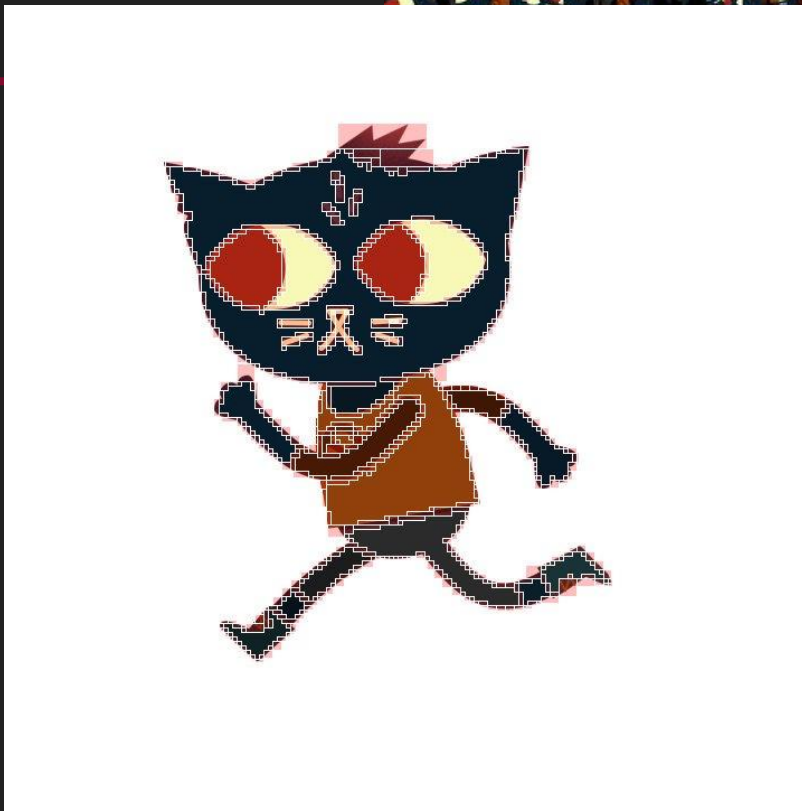


“Tile engines give a **structured** way to create large worlds with a very small amount of graphic assets. They encourage **asset reuse** by breaking the world up into a grid that can have tiles placed into each grid cell.”

An Introduction to HTML5 Game Development





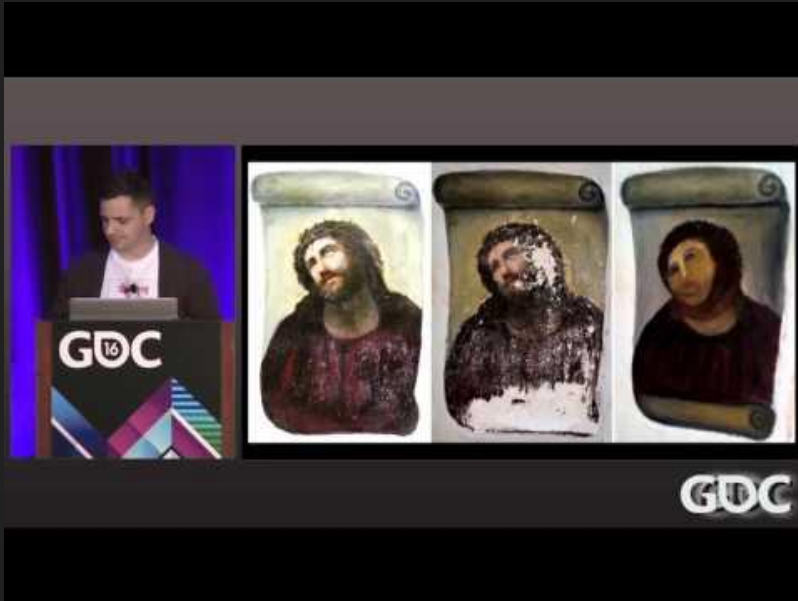


By Grabthar's Hammer, What a Savings: Making
the Most of Texture Memory with Sprite Dicing
By Jon Manning
<https://www.gdcvault.com/play/1025419/By-Grabthar-s-Hammer-What>



https://www.gamasutra.com/blogs/JoelBurgess/20130501/191514/Skyrims_Modular_Approach_to_Level_Design.php

Modular Level Kits



http://twvideo01.ubm-us.net/ol/vault/gdc2016/Presentations/Burgess_Joel_Modular%20Level%20Design.pdf

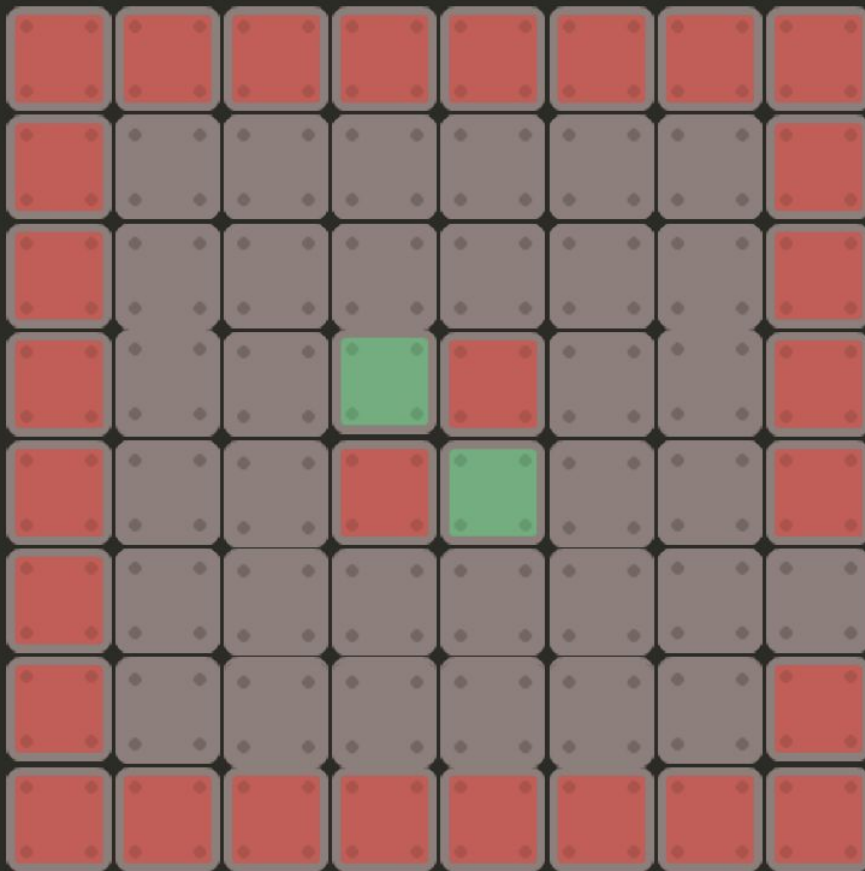
<https://www.youtube.com/watch?v=QBAM27YbKZg>



<https://youtu.be/JKn7u09mR8M>

Tiles are efficient

```
var map = [  
  [1, 1, 1, 1, 1, 1, 1, 1],  
  [1, 0, 0, 0, 0, 0, 0, 1],  
  [1, 0, 0, 0, 0, 0, 0, 1],  
  [1, 0, 0, 2, 1, 0, 0, 1],  
  [1, 0, 0, 1, 2, 0, 0, 1],  
  [1, 0, 0, 0, 0, 0, 0, 0],  
  [1, 0, 0, 0, 0, 0, 0, 1],  
  [1, 1, 1, 1, 1, 1, 1, 1]  
];
```



“**Tiled** is a general purpose tile map editor. It is meant to be used for editing maps of any tile-based game, be it an RPG, a platformer or a Breakout clone.”

Tiled documentation



Tiled Map Editor

<https://www.mapeditor.org/>

Tiled is a free software level editor. It supports editing tile maps in various projections (orthogonal, isometric, hexagonal) and also supports building levels with freely positioned, rotated or scaled images or annotating them with objects of various shapes.

Even though Tiled is available for free, I accept voluntary payments in order to be able to spend more time on it. I'm currently spending two full days/week on Tiled, which is possible thanks to people choosing to pay for Tiled here as well as those supporting me on a recurring basis through Patreon.

I did not quite reach my funding goal yet, so if you enjoy using Tiled and are able to chip in, please set up a [small monthly donation through Patreon](#). Thanks!

[More information](#) ▾

Download


[Download Now](#) Name your own price

Click download now to get access to the following files:

Tiled for Windows (32-bit), installer 20 MB 
Version 1.2.3


Tiled for Windows (64-bit), installer 16 MB 
Version 1.2.3

Tiled for macOS 14 MB 
Version 1.2.3

Tiled for Linux (64-bit), release 26 MB 
Version 1.2.3

Tiled for Windows (32-bit), snapshot 20 MB 
Version 2019.04.25

Tiled for Windows (64-bit), snapshot 18 MB 
Version 2019.04.25

Tiled for Windows XP, snapshot 19 MB 
Version 2019.04.25

“Tiled supports **straight rectangular** tile layers, but also **projected isometric**, **staggered isometric**, and **staggered hexagonal** layers.

A tileset can be either a single image containing many tiles, or it can be a collection of individual images.

In order to support certain **depth-faking techniques**, tiles and layers can be offset by a custom distance and their rendering order can be configured.”

Tiled Documentation

USER MANUAL

Introduction

Working with Layers

Editing Tile Layers

Working with Objects

Editing Tilesets

Custom Properties

Using Templates

Using the Terrain Brush

Using Wang Tiles

Using Infinite Maps

Using Commands

Automapping

Export Formats

Keyboard Shortcuts

User Preferences

Python Scripts

REFERENCE

Libraries and Frameworks

TMX Map Format

TMX Changelog

JSON Map Format

Tiled Documentation

Note

If you're not finding what you're looking for, ask questions on the [Tiled Forum](#).

User Manual

- Introduction
 - [About Tiled](#)
 - [Getting Started](#)
- Working with Layers
 - [Tile Layers](#)
 - [Object Layers](#)
 - [Image Layers](#)
 - [Group Layers](#)
- Editing Tile Layers
 - [Stamp Brush](#)
 - [Terrain Brush](#)
 - [Wang Brush](#)
 - [Bucket Fill Tool](#)
 - [Shape Fill Tool](#)
 - [Eraser](#)
 - [Selection Tools](#)
 - [Managing Tile Stamps](#)
- Working with Objects
 - [Placement Tools](#)
 - [Select Objects](#)
 - [Edit Polygons](#)
- Editing Tilesets
 - [Two Types of Tileset](#)
 - [Tileset Properties](#)

New tilemap

The 'New Map' dialog box is shown with the following settings:

- Map**
 - Orientation: Orthogonal
 - Tile layer format: CSV
 - Tile render order: Right Down
- Map size**
 - ☒ Fixed
 - Width: 30 tiles
 - Height: 20 tiles
 - 960 x 640 pixels
 - ☐ Infinite
- Tile size**
 - Width: 32 px
 - Height: 32 px

Buttons: Cancel, Save As...

straight rectangular tile layers

CSV = comma-separated values

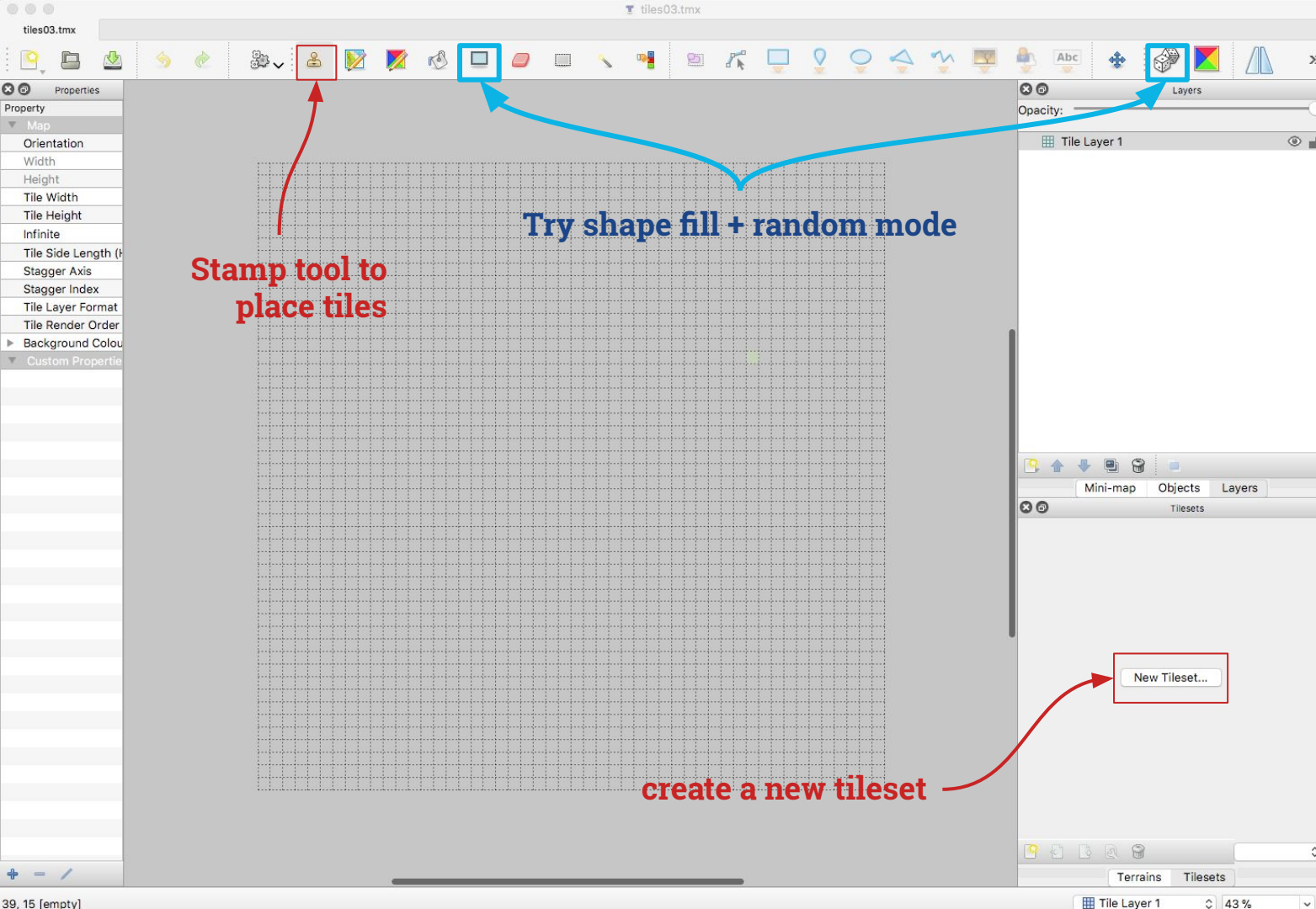
z-index rendering order for overlapping tiles

tile width

tile height

[Best to stick with powers of 2]

map size in tiles



Create a new tileset (will save a .tmx file)

Tileset name →
What kind of tileset are you
creating? (Make sure to "Embed in map") →

Which image are you using? →

Which color is transparent? →

New Tileset - Tiled

Tileset

Name:

Type: Based on Tileset Image ☐ Embed in map

Image

Source: Browse...

☐ Use transparent color:

Tile width: 32 px Margin: 0 px

Tile height: 32 px Spacing: 0 px

Save As... Cancel

Margin = space *around* tiles
Space = space *between* tiles
(both of these are usually 0)

Tiled Layer Types

Tile Layers: Arrays of tile reference data. Contain no additional information beyond tile orientation flags. All tiles are grid-aligned.

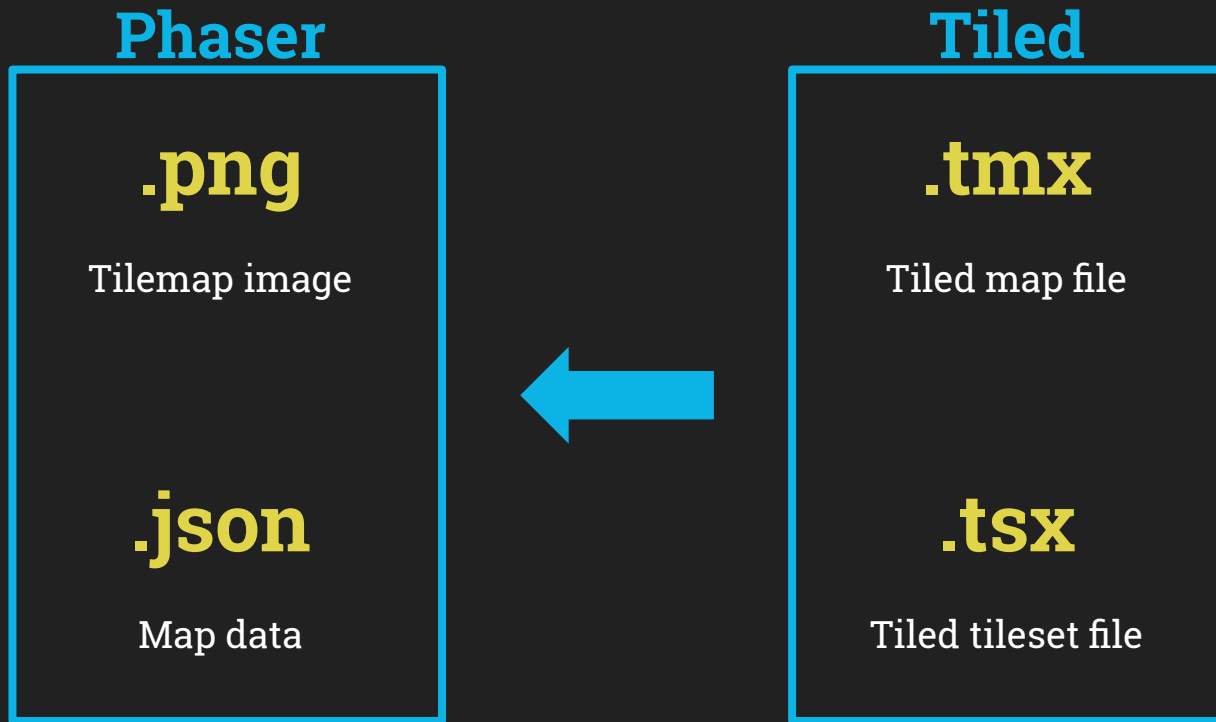
Object Layers: Store additional information that doesn't fit in Tile Layers. Objects may be freely positioned, resized, and rotated. Allows for custom properties.

Group Layers: Used to group and organize layers into hierarchies.

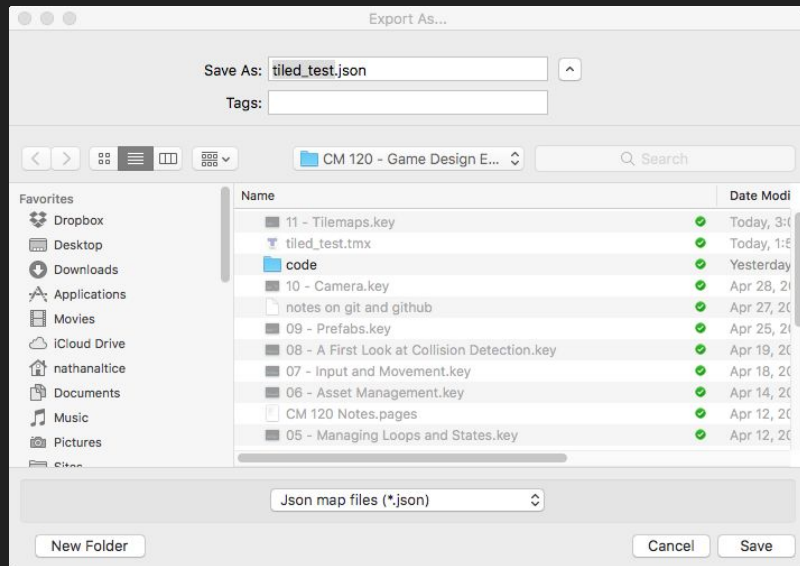
Image Layers: Contain a single image for game foregrounds and/or backgrounds. (Not particularly useful.)

Note: All layers may be hidden, partially visible, or given an offset to create depth effects.

File Formats



Export to Phaser



You'll need to export in ".json" map files format!

[illegible]

Tilemaps in Phaser

1. Load the data from Tiled (both the JSON data and the spritesheet)
2. Create a Phaser **Tilemap** object
3. Add a fileset image to the **Tilemap** object
4. Define **Tilemap** collision
5. Create a **Tilemap Layer** that combines image with data
6. Resize the game world to your **Tilemap Layer**
7. Check object/map collisions

Tilemaps in Phaser

preload

1. Load the data from Tiled (both the JSON data and the spritesheet)

2. Create a Phaser **Tilemap** object

3. Add a fileset image to the **Tilemap** object

4. Define **Tilemap** collision

5. Create a **Tilemap Layer** that combines image with data

6. Resize the game world to your **Tilemap Layer**

7. Check object/map collisions

create

update

```

1. Play.prototype = {
    preload: function() {
        game.load.path = 'assets/';
        // load tilemap data (key, url, data, format)
        game.load.tilemap('level', 'tiled_level.json', null, Phaser.Tilemap.TILED_JSON);
        // load tilemap spritesheet (key, url, frameWidth, frameHeight)
        game.load.spritesheet('tilesheet', 'tilesheet_complete.png', 32, 32);

        create: function() {
            // spin up physics
            game.physics.startSystem(Phaser.Physics.ARCADE);
            game.physics.arcade.gravity.y = this.GRAVITY;
            // TILE_BIAS adds a pixel "buffer" around your tiles to avoid collision tunneling
            // see https://thoughts.amphibian.com/2016/02/dont-fall-through-tile-bias-in-phaser.html
            game.physics.arcade.TILE_BIAS = 32;

            // set bg color
            game.stage.backgroundColor('#87CEEB');

            // create new Tilemap object - when using Tiled, you only need to pass the key
2. this.map = game.add.tilemap('level');
            // add an image to the map to be used as a tilesheet (tilesheet, key)
            // the tilesheet name is specified w/in the .json file (or in Tiled)
            // a single map may use multiple tilesheets
3. this.map.addTilessetImage('abstract platformer', 'tilesheet');
            // set ALL tiles to collide *except* those passed in the array
4. this.map.setCollisionByExclusion([]);
            // create new TilemapLayer object
            // A Tilemap Layer is a set of map data combined with a tilesheet
5. this.mapLayer = this.map.createLayer('Tile Layer 1');

            // set the world size to match the size of the Tilemap layer
6. this.mapLayer.resizeWorld();

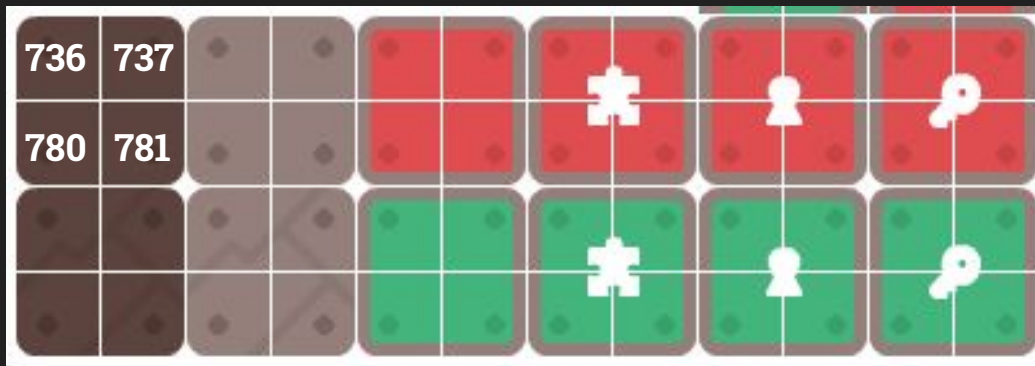
            update: function() {
                // collision checks
7. game.physics.arcade.collide(this.player, this.mapLayer);
            }
        }
    }
};

```

Tilemap collision options

```
Tilemap.setCollision([736, 737, 780, 781], true);
```

Set the collision for specific tile indices within your Tilemap object. (That last boolean enables collision.)



Tilemap collision options

```
Tilemap.setCollisionBetween(794, 799, true);
```

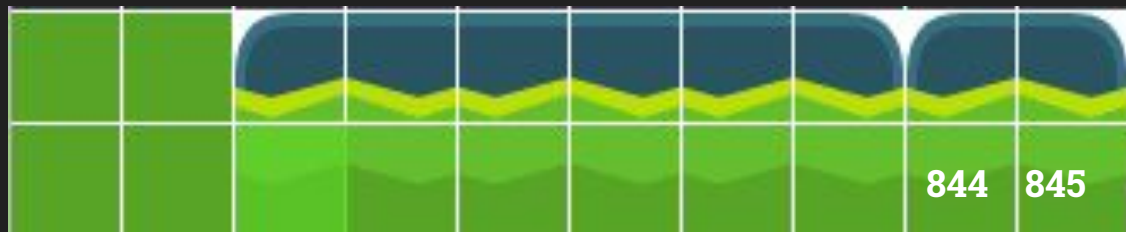
Set the collision for a range of sequential tiles within your Tilemap object.



Tilemap collision options

```
Tilemap.setCollisionByExclusion([844, 845]);
```

Set the collision for *all* within the Tilemap *except* for those specified in the passed array.





COLLISION WEIRDNESS



If you are having collision problems between objects (e.g., your player and map tiles), be sure you're using **physics** to propel objects rather than direct x-/y-coordinate control.

For some reason, `body.touching` does not work for sprite/tilemap collisions, but `body.blocked` does. *I do not know why.*

If you're having collision tunneling issues because your objects move at high speeds, increase the **TILE_BIAS** to a value higher than the default 16 px.

Play.prototype = {}

```
create: function() {  
    // spinup physics  
    game.physics.startSystem(Phaser.Physics.ARCADE);  
  
    // set bg color  
    game.stage.setBackgroundColor('#87CEEB');  
  
    // create new Tilemap object - when using Tiled, you only need to pass the key  
    map = game.add.tilemap('level');  
    // add an image to the map to be used as a tileset (tileset, key)  
    // the tileset name is specified w/in the .json file (or in Tiled)  
    // a single map may use multiple tilesets  
    map.addTilesetImage('tilesheet_complete', 'tilesheet');  
    // create new TilemapLayer object  
    // A Tilemap Layer is a set of map data combined with a tileset  
    decorationLayer = map.createLayer('Decoration');  
    terrainLayer = map.createLayer('Terrain');  
  
    // set ALL tiles to collide *except* those passed in the array  
    map.setCollisionByExclusion([]);  
    // set the world size to match the size of the Tilemap layer  
    terrainLayer.resizeWorld();
```

Multiple layers
(but only one can
have collision)

```
// setup keys group  
keys = game.add.group();  
keys.enableBody = true;
```

Sprites from objects

```
// convert Tiled objects w/ xx ID into key sprites  
// createFromObjects creates a sprite for every object matching the given gid (grid ID) in  
// (object group name, gid, image key, frame, exists, autocull, group)  
map.createFromObjects('Keys', 1057, 'key', 0, true, true, keys);
```

Adding Object Layers?

Important: Tiled's Object Layer is simply a visual representation of tile references. In Phaser, this data must be converted into display objects. Adding an Object Layer with `.createLayer()` will not work.

```
index.html x tilemap01.js x tilemap02.js
{
  "draworder": "topdown",
  "height": 20,
  "name": "Keys",
  "objects": [
    {
      "gid": 1057,
      "height": 30,
      "id": 17,
      "name": "",
      "rotation": 0,
      "type": "",
      "visible": true,
      "width": 29,
      "x": 1264.6666666666667,
      "y": 369.6666666666667
    },
    {
      "gid": 1057,
      "height": 30,
      "id": 18,
      "name": "",
      "rotation": 0,
      "type": "",
      "visible": true,
      "width": 29,
      "x": 1774,
      "y": 236.33333333333333
    },
    {
      "gid": 1057,
      "height": 30,
      "id": 19,
      "name": "",
      "rotation": 0,
      "type": ""
    }
  ]
}
```

Line 1, Column 1

Tilemaps are tricky in Phaser!

Be sure to consult both the Phaser CE [Tilemaps documentation](#) and the [Tilemaps examples](#) for more info.

Class: Tilemap

Phaser. Tilemap

new Tilemap(game [, key] [, tileWidth] [, tileHeight] [, width] [, height])

Creates a new Phaser.Tilemap object. The map can either be populated with data from a Tiled JSON file or from a CSV file.

Tiled is a free software package specifically for creating tile maps, and is available from <http://www.mapeditor.org>

To do this pass the Cache key as the first parameter. When using Tiled data you need only provide the key. When using CSV data you must provide the key and the tileWidth and tileHeight parameters. If creating a blank tilemap to be populated later, you can either specify no parameters at all and then use `Tilemap.create` or pass the map and tile dimensions here. Note that all Tilemaps use a base tile size to calculate dimensions from, but that a TilemapLayer may have its own unique tile size that overrides it. A Tile map is rendered to the display using a TilemapLayer. It is not added to the display list directly itself. A map may have multiple layers. You can perform operations on the map data such as copying, pasting, filling and shuffling the tiles around.

Parameters:

Name	Type	Argument	Default	Description
<code>game</code>	<code>Phaser.Game</code>			Game reference to the currently running game.

Your endless runners

More Debugging Tips

Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
 - a. `assert()`
 - b. Keep a debugging notebook
2. Make debug tools
 - a. Quicker feedback is better
 - b. Display values live if possible
3. Only make one change at a time and then test it
4. Just because you paused the game doesn't mean it's paused
 - a. And stopping one update doesn't mean you stopped all of them
5. `console.log()` is slow
 - a. Faster to print an array as a string than to individually print the contents

Useful random debugging advice

Walk through your code step by step, explaining to yourself what is supposed to happen

Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
 - a. `assert()`
 - b. Keep a debugging notebook
2. Make debug tools
 - a. Quicker feedback is better
 - b. Display values live if possible
3. Only make one change at a time and then test it
4. Just because you paused the game doesn't mean it's paused
 - a. And stopping one update doesn't mean you stopped all of them
5. `console.log()` is slow
 - a. Faster to print an array as a string than to individually print the contents

Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
 - a. `assert()`
 - b. Keep a debugging notebook
2. Make debug tools
 - a. Quicker feedback is better
 - b. Display values live if possible
3. Only make one change at a time and then test it
4. Just because you paused the game doesn't mean it's paused
 - a. And stopping one update doesn't mean you stopped all of them
5. `console.log()` is slow
 - a. Faster to print an array as a string than to individually print the contents

Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
 - a. `assert()`
 - b. Keep a debugging notebook
2. Make debug tools
 - a. Quicker feedback is better
 - b. Display values live if possible
3. Only make one change at a time and then test it
4. Just because you paused the game doesn't mean it's paused
 - a. And stopping one update doesn't mean you stopped all of them
5. `console.log()` is slow
 - a. Faster to print an array as a string than to individually print the contents

Useful random debugging advice

1. When you find a problem, change something so that same problem can't happen again
 - a. `assert()`
 - b. Keep a debugging notebook
2. Make debug tools
 - a. Quicker feedback is better
 - b. Display values live if possible
3. Only make one change at a time and then test it
4. Just because you paused the game doesn't mean it's paused
 - a. And stopping one update doesn't mean you stopped all of them
5. `console.log()` is slow
 - a. Faster to print an array as a string than to individually print the contents

AABB characters and slopes

An example of a real-world
physics-and-debugging problem in a game
with 2D physics like yours

<https://twitter.com/eevee/status/1133248372624613376>