PHASER WORLD

APRIL 2017 ISSUE 76

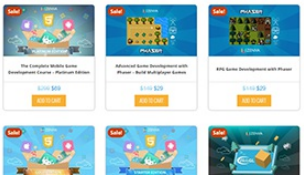
CONTENTS

Welcome to Issue 76 of Phaser World

In many parts of the world this issue is likely to arrive over the Easter holidays. If that's you, I hope you are enjoying your nice extended weekend. If you fancy a bit of game time then we've got some great titles this issue for you (Hop Til You Drop being quite suitable), or if you're looking forward to using the time to get some game dev done, then we've a bunch of new tutorials too.

Someone reached out to me on twitter recently asking how they unsubscribe from the newsletter. If you scroll to the very bottom, you'll see a link in the footer to do exactly this. If you can't see the footer because your mail client has truncated the newsletter, then you'll need to view the email in full to see the link. I don't mind at all if people want to unsubscribe, because it actually costs me money each month to send this to you. So I'd much rather only interested people receive it. That said, please do stick around if you're even remotely interested in Phaser, because we've got some big events coming up!

Until the next issue, keep on coding. Drop me a line if you've got any news you'd like featured (you can just reply to this email) or grab me on the Phaser Slack or Discord channels.



Zenva are having a huge end of season sale. Their popular courses like Advanced Phaser Game Dev Course, Phaser RPG Game Dev Course, Mobile Game Dev Course, and many others at massively reduced prices - some down from $150 to $29. Check it out before it ends! If you buy a course via that link then a % goes towards Phaser development too.

Games made with Phaser

Cartoon Candies

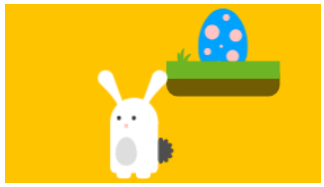
Game of the Week
This match 3 game has beautiful graphics, bouncy animation, and is dripping with sugary charm.

Phaser Quest

Staff Pick
Explore the world, find better equipment, fight monsters and defeat the final boss - alone or with friends.

Loon Ride

Continue flying upwards, without touching the cactus, in this epic and addictive hot air balloon game!



Hop 'Til You Drop

Collect the eggs, avoid the foxes, and see how high you can get in this super-cute Easter game.

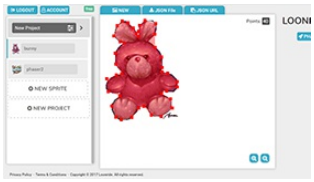
Crazy Eggs

An Easter themed shuffle game. Keep your eye on the eggs as they zoom around.

Phaser News & Tutorials

Phaser CE v2.7.6 Released

The latest version of Phaser CE is now released, including lots of fixes, updates and new features.



Loon Physics

Loon Physics is a P2 Physics Body Editor, that exports the physics data to json for Phaser.

Loon Physics Tutorial

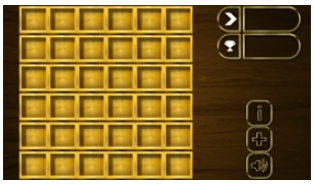
How to make a game using P2 physics to create non-rectangular collision.

State Transition Tutorial

A tutorial on how to smoothly transition from one Phaser State to another.

[Latency Estimation Tutorial]

An article explaining how the server keeps track of the latency of all connected players in Phaser Quest.



[Responsive Game Tutorial Part 3]

The final tutorial in the series on making responsive games, this time offering different layouts per orientation.

Patreon Updates

Thank you and welcome to the following awesome people who joined the Phaser Patreon this week: **Tony Jennings** and **Samid**.

Patreon is a way to contribute towards the Phaser project on a monthly basis. This money is used *entirely* to fund development costs, and is the only reason we're able to invest so much time into our work. You can also donate via PayPal.

Backers are entitled for free monthly support from me via Slack or Skype, as well as discounts on plugins and books, and forum badges.

Development Progress

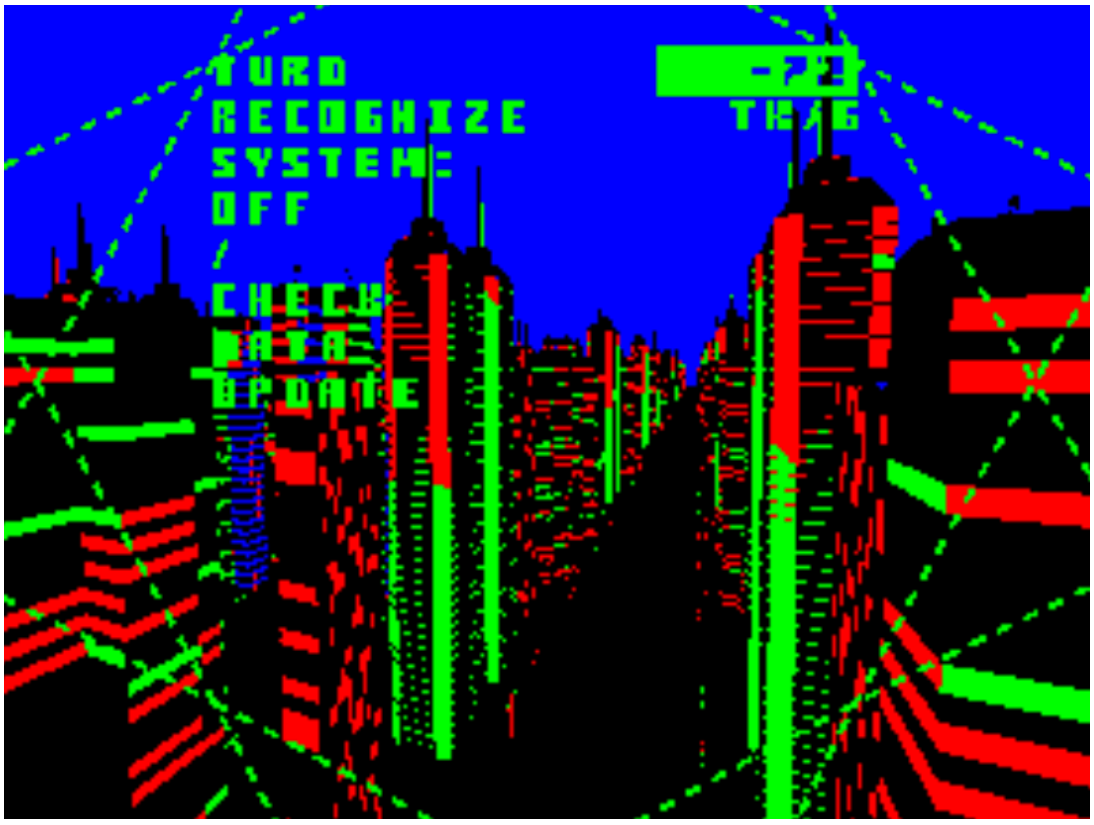
The community has pulled together and done some sterling work on Phaser CE, and I'm pleased to say that Phaser CE v2.7.6 is now available. It has been published to npm and released on GitHub (the CDNs should follow soon), so however you like to grab it, you now can! There are loads of fixes, updates and features in this release, and it's amazing to see it being adopted by the community like this.

Phaser 3 Development

Last week I talked about the work I had started on the new Animation system, and this week I continued with that, along with a diversion into the land of JSON. First let's discuss some of the new features of the Phaser 3 Animation Manager, as usual you can click the screen shots to run the demos.

Animations Over Multiple Texture Atlases

You won't believe how often this is asked for in Phaser 2 - the ability to define an animation that has frames from more than one texture atlas. In v2 it's basically impossible (without some major hacking of the core), so I was adamant it would "just work" in v3, and now it does:



The above animation has 99 frames, each 320x240 in size, and is split over 3 texture atlases. The entire example is just 4 lines of code (excluding function wrappers).

Along similar lines I tested converting a small piece of video into an animation sequence (I actually converted it to an animated GIF first, to get the sequence right, then exported to frames). As it's just rendering a texture, and not an actual video, it was easy to pull off some fun effects on it, such as swirling it about, and leaving an alpha trail behind:

Animation Callbacks

It's very often a requirement to be able to link other game events to animations. For example if a player death animation plays out, when it completes you likely want to trigger a Game Over sequence. Or perhaps you need a callback as an animation is playing, on each frame change.

To accommodate this, animations now have callbacks for: on start, on update, on repeat and on complete. The callbacks can be defined on the animation itself, or on an instance of the animation. Remember under v3 that animations are now global, able to be shared between multiple game objects. So you can set the 'global' animation to trigger the callbacks, or you can assign an animation to a Game Object, and then set the callbacks just on that. It gives you more flexibility, but in the default set-up uses less memory. The way you define these callbacks is via the Animation Configuration object, which leads us nicely onto the next section.

Phaser 3 Configuration Objects

In Phaser 2 pretty much everything is defined via parameters. Some methods have a huge long list of parameters, and remembering them can be a nightmare, especially if you don't have code completion in your editor. In v3 we still do this for anything that is really hot, i.e. is likely to be called in a core game loop, or requires internal optimization by v8 and similar compilers. But when you're defining an animation, or even a Sprite, that isn't usually a hot event - it's typically more a set-up, or one-off event, not something many hundreds of times a second.

Because creating objects is when you need a more verbose and easier to parse approach we're using proper configuration objects across all of v3. This week I added support for it into the Animations, and also into Game Objects themselves. Here is a basic config object for defining an animation:

```
var config = {
    key: 'walk',
    frames: this.anims.generateFrameNumbers('mummy'),
    framerate: 6,
    yoyo: true,
    repeat: -1
};

this.anims.create(config);
```

I'm sure most of you can figure out what this config is going to do, even having never touched Phaser 3, or used the API yourselves. 'this.anims' is the global Animation Manager, available from within any Phaser State. The above snippet is defining an animation, using a loaded sprite sheet (called 'mummy'). It has a framerate of 6 fps, will yoyo during playback, and will repeat forever (the -1 value). Had repeat been a positive integer it would have repeated for that number instead. Had it been left out entirely, it would just play through once.

Here is the full Animation configuration, showing all options currently available (it will extend over time):

```
frames: [
    { key: textureKey, frame: textureFrame },
    { key: textureKey, frame: textureFrame, duration: float },
    { key: textureKey, frame: textureFrame, visible: boolean },
    { key: textureKey, frame: textureFrame, onUpdate: function }
],
framerate: integer,
duration: float (seconds, optional, ignored if framerate is set),
skipMissedFrames: boolean,
delay: integer,
repeat: integer (-1 = forever),
repeatDelay: integer,
yoyo: boolean,
showOnStart: boolean,
hideOnComplete: boolean,
callbackScope: Object,
onStart: function,
onStartParams: array,
onRepeat: function,
onRepeatParams: array,
onUpdate: function,
onUpdateParams: array,
onComplete: function,
onCompleteParams: array,
transitions: [
    {
        key: string <- key of the animation to blend with,
        frames: [] <- play these frames before starting key
    }
]
```

Now as well as being able to define an animation like this, you can also save these configs to JSON files and use the new Loader.animation call to read them back in again. This means you can define all of your animations in an external JSON file, and have Phaser load them as part of its preload process. When it finds an animation file it will automatically add those animations into the Animation Manager

You can see an example of this here: Load Animation JSON

Export Animation JSON

As well as loading JSON data, you can also export it from within v3. The Animation Manager, Animations and Animation Frames all have native 'toJSON' functions now. As you may know, defining a function on a prototype called 'toJSON' allows you to pass the object directly to JSON.stringify, and it will natively call your own function. This allows you to ensure only valid data is exported, and avoid circular references:

```
//  Get a JSON representation of a single animation,
//  or all animations:

//  You can extract the animation:
var ruby = this.anims.get('ruby');

//  Then pass it to JSON.stringify
JSON.stringify(ruby);

//  Or call toJSON directly (this returns an Object)
ruby.toJSON();

//  You can also call 'this.anims.toJSON'
//  and pass it the key of the animation you want:
JSON.stringify(this.anims.toJSON('ruby'));

//  Or dump out ALL animations in the Animation Manager:
JSON.stringify(this.anims);
```

I feel that the ability to easily export and import JSON data like this will have multiple benefits: First it means that your animators can try new things out without even touching the game code. Secondly it makes your code cleaner, as you don't need to spend many lines configuring animations. And third it hopefully allows for a new range of 3rd party tools to arise, that export to Phasers Animation format.

Bone Based Animations

Don't get too excited, as work on this hasn't started yet. But I just wanted to mention that I've been really careful to ensure that all of the animation work done so far is self-contained. I'm calling them 'frame based animations' internally, and all the code is confined to its own folder and objects. This is so we can easily slot in bone / skeleton animation formats in the future easily, without hijacking existing classes. Current animations know they are frame based only, and are treated as such internally. My plan is to support multiple format in the future (Spriter, Spine, Creature, DragonBones), and not be confined to just one, so it has all been designed to make that easy to do.

JSON Everywhere

The work I've been doing with animation configurations didn't end there. I also spent a good chunk of time doing the same with Phaser Game Objects (such as Sprites, Text, Layers, etc). This means you can now configure game objects from JSON data, as well as their animations. There is a global Game Object 'build from configuration' function that is in v3, and then each specific type of object extends from that, adding in any extra parts it needs.

As with animations, Game Objects now have native 'toJSON' functions too, and I have modified the Game Object Creator to use purely config object. This means you now have a choice: You can use 'this.add.sprite' and create a Sprite in the normal way, using parameters. Or you can now use 'this.make.sprite' and pass in a config object instead. The configuration objects allow for some really smart features to be exposed, but I've already talked for long enough this issue - so I'll cover it in more depth next week.

Getting ever closer

If you've been following along for the past few weeks I hope you can see the difference it has made having me dedicated to v3 full-time. It really feels like we've got a roaring fire burning under the project, and new areas are coming online weekly.

Sadly though, a couple of weeks ago we lost our biggest patron, which put a huge $600 dent in our monthly total. These things happen, and I know circumstances change, but I'll be honest - the timing was painful! I've been pruning back expenses elsewhere to try and mitigate this, and thankfully we've had some new patrons since then too (thank you!) - but all I'd ask is *if you're able*, please keep your support going for the next couple of months.

I appreciate that it's hard because v3 isn't actually out yet, but we get ever closer every week, and it honestly feels like we're only a couple of months away from a good stable, production worthy build. As soon as Phaser 3 is out I can turn my

focus to creating books and learning materials, which will generate income, so we're not as reliant on Patreon, but right now we're in the midst of the battle and I can't take my eyes off the goal. If you support us already, I really do thank you, and I truly hope you'll be as pleased with v3 as I am - and please hold in there for a little longer if you can! If you don't support us and would like to, believe me, now is the time. Even if you only do it for a couple of months and then cancel, that is fine too! It all helps.

Phaser 3 Mailing List and Developers Guide

If you're interested in helping evolve the shape of Phaser 3, then please join the Phaser 3 Google Group. Discussions this week have included varying render loops. The group is for anyone who wishes to help shape what the Phaser 3 API and feature-set will contain.

The Phaser 3 Developers Guide is available. Essential reading for anyone who'd like to help *build* Phaser 3.

Geek Links

I loved this Super adventures in Amiga disk screens article - so many classic games, all asking you to do the same thing: Insert disk B :)

Read about the remarkable community around the 27-year old MS-DOS racing game: Stunts.

It has to be said, the trailer for Thor: Ragnarok looks pretty awesome! I dig the 80s vibe it's got going on there.

Got a suggestion for a Geek Link? Email it to me (support@phaser.io)

Phaser Releases

The current version of Phaser CE is 2.7.6 released on April 13th 2017.

Phaser 3.0.0 is in active development in the GitHub v3 folder.

Please help support Phaser development

Have some news you'd like published? Email support@phaser.io or tweet us.

Missed an issue? Check out the Back Issues page.