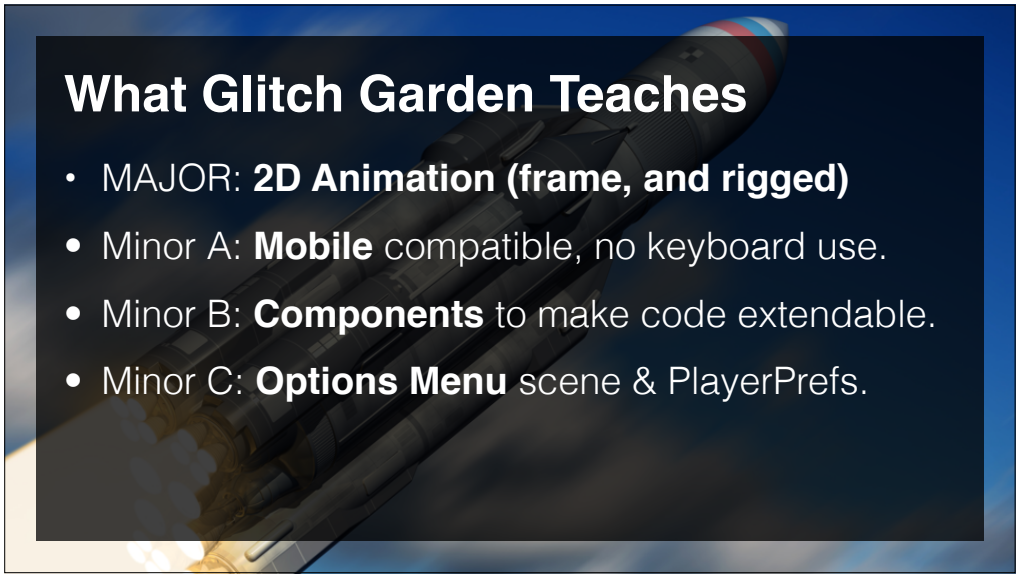
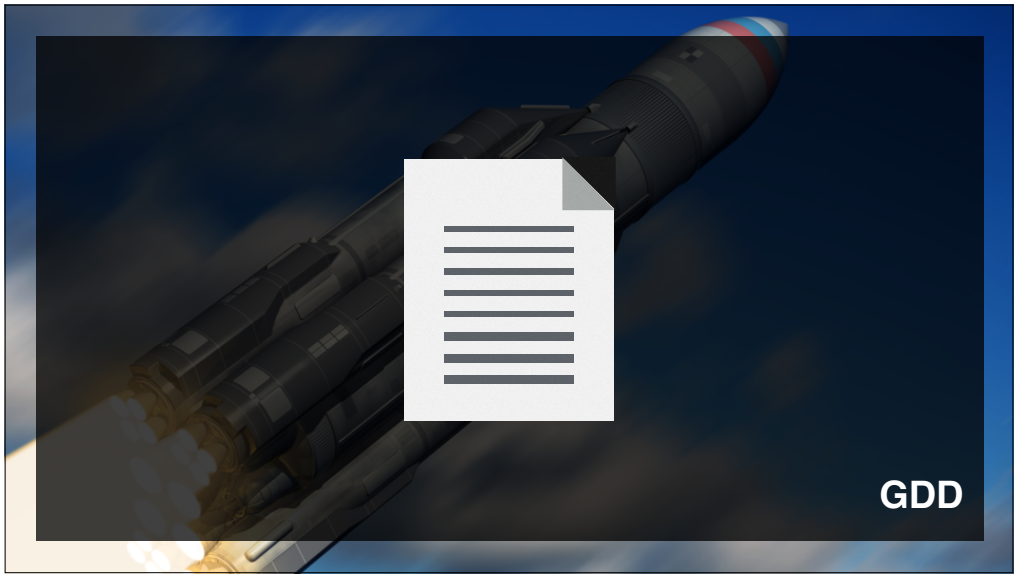
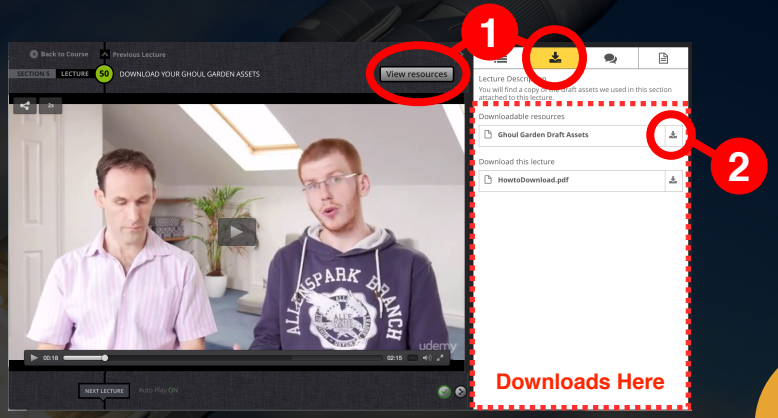

ACT I: Creating The Context
@UnityCourse
facebook.com/UnityCourse

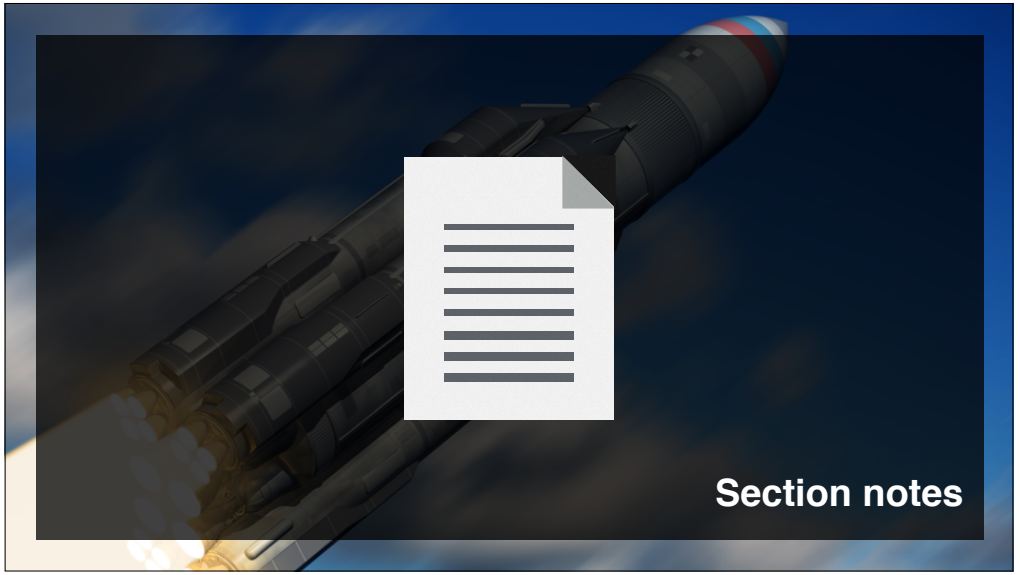

About Ghoul Garden


What Glitch Garden Teaches
• MAJOR: 2D Animation (frame, and rigged)
• Minor A: Mobile compatible, no keyboard use.
• Minor B: Components to make code extendable.
• Minor C: Options Menu scene & PlayerPrefs.


GDD

Your Glitch Garden Assets





Testing Our Destruction

	One Zombie	Zombies Bunched	Sequential Zombies
Plant Dies First			
A Zombie Dies First			
All Bunched Zombies Die			

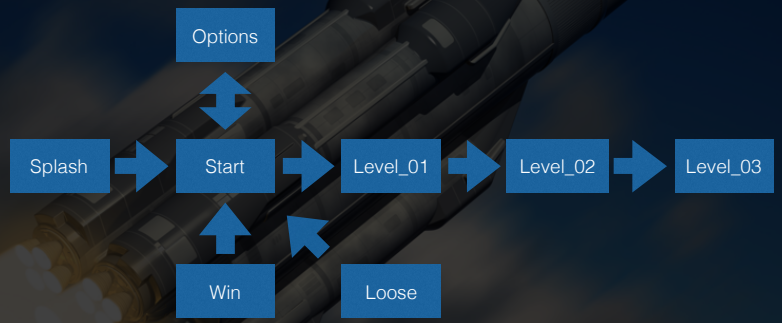
In this video…

- What is a splash screen
- Why use a splash screen
- Singleton-free music manager
- Make you splash-screen
- Add music, and make Start Menu auto-load

Your splash screen

Unity's Mandatory Splash Your Splash Your Start Menu

Singleton-free music manager



Some resources

- Fonts: www.dafont.com
- Music: www.freesound.org
- Images: Google images etc.
- Also, wallpaper and music from www.glitchthegame.com/downloads

Make you splash-screen

- Add a simple UI > Image*.
- Add a short start sound (5s or less).
- Import Start Scene from previous project.
- Arrange for Start Scene to load after 1-5s

** Don't worry about scaling for now, suggest 16:9*

In this video…

- We're building for "mobile first" here.
- No use of keyboard, just tap & drag.
- Mobile device aspect ratios.

Mobile Device Aspect Ratios

Aspect Ratio	Decimal	Example Devices
4:3	1.3	iPad (1, 2 & 3). Some old Android devices.
3:2	1.5	iPhone (<= 4S). Some Android devices.
16:10	1.6	Android phones and tablets.
17:10	1.7	Android tablets
16:9	**1.8**	**iPhone 5 / 6. Android devices**

http://rusticode.com/2014/01/11/handling-resolutions-and-aspect-ratio-of-common-mobile-devices-for-web-application-and-game-development

Background Image Options

- Stretch: easy but ugly.
- Black bars: appropriate in extreme cases.
- Crop edges: ok with care about what's lost.

Quiet Zones On Image Sides

Setup Start Scene

- Read Unity Docs*
- Explore various options
- Try and get a satisfactory result

http://docs.unity3d.com/Manual/class-RectTransform.html

http://docs.unity3d.com/Manual/script-CanvasScaler.html

Alternative Music Manager

@UnityCourse

facebook.com/UnityCourse

In this video…

- An alternative MusicManager.cs architecture
- Customise your Win and Loose scenes.
- Test it all looks and sounds good.

Our level structure

Setup Your Menu System

- Customise Win and Loose Scenes.
- Add a new Options scene (blank for now).
- Add two buttons: “Back”, “Defaults”.
- Make Level_01 with “Win” and “Loose” buttons.
- Test all the navigation and music works properly.

Menus, Options & Music

@UnityCourse
facebook.com/UnityCourse

In this video…

- Customise Win and Loose Scenes.
- Add a new Options scene (blank for now).
- Add two buttons: “Back”, “Defaults”.
- Make Level_01 with “Win” and “Loose” buttons.
- Test all the navigation and music works properly.

Why We Prefab LevelManager

- Need instance to use UI OnClick().
- With persistent scripts, isn’t in scene at edit time.
- **Can** use a prefab, but then can’t properties.
- Methods can’t be static for use with OnClick().
- So prefab “Level Manager”, optionally drop in.

In this video…

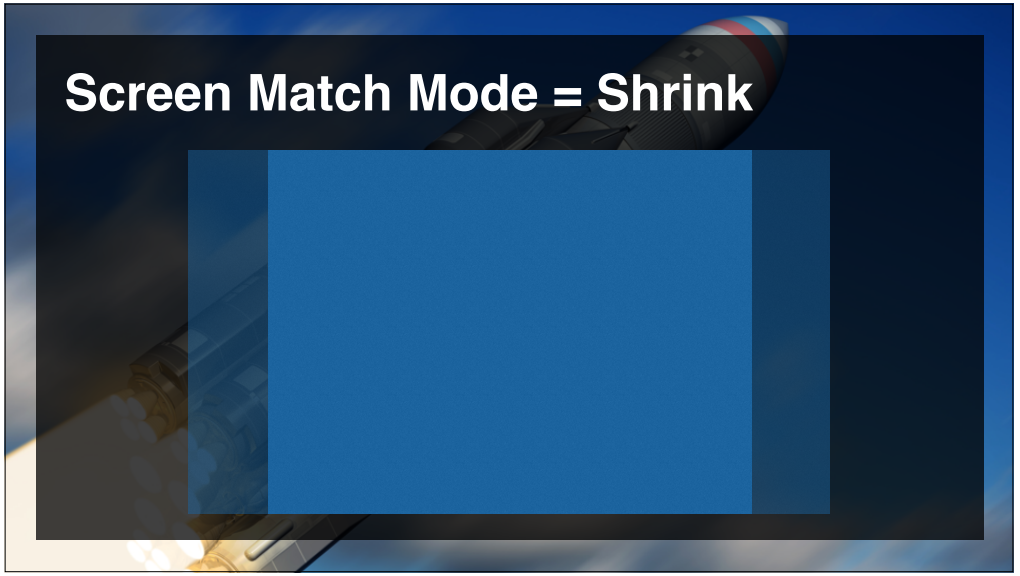
- Adding a nice fade-in to the Start Scene.
- Giving-up on spelling loose / lose / whatever.
- Add background image to levels.
- Check it all flows / scales nicely.

Make Start Scene Fade In

- Make the start scene fade-in over a variable time.
- Try it your way, it's great learning.
- Use Google / think of alternatives.
- Hint 1: I'll be using a UI > Panel and it's "alpha".
- Hint 2: A good place for the script is this panel.

In this video…

- Canvas Scaler "Screen Match Mode".
- Use a "Raw Image" & grass texture.
- Define play space, and quiet zones.
- Setup our Level with prefabs.



Add Grass To Levels

- At least 5 rows and 9 columns at 4:3.
- Make it fill the screen for now.
- Prefab the Canvas for other levels.
- Ensure it scales “properly” with Game window.

In this video…

- What is PlayerPrefs, and why is it useful?
- Limitations of PlayerPrefs
- Why we’re providing our own wrapper class.
- Create PlayerPrefsManager.cs static wrapper.

Limitations of PlayerPrefs

- Can be slow to read and write.
- Max file size is 1 MB.
- Only supports Float, Int, String (no bool etc)
- Dont’ abuse!

Our Wrapper Class

- Centralises all PlayerPrefs read / write.
- Static so available from anywhere.
- Allows for checking / error handling.
- Much safer and clearer.

Write IsLevelUnlocked (int level)

- Returns true if the value against the key is 1.
- Returns false otherwise.
- If level index not in build order…
  - Log a helpful error.
  - Return false.

Write Difficulty Wrapper

- const string DIFF_KEY = “difficulty"
- public static void SetDifficulty (float volume)
- public static float GetDifficulty ()
- check it all works with Debug.Log

In this video…

- Introducing UI sliders.
- Add volume and difficulty sliders.
- Create OptionsController.cs.
- Ensure sliders work.

Get Difficulty Options Working

- Change PlayerPrefsManager.cs check.
- Allow values of 1 <= difficulty <= 3.
- Make slider “whole numbers”, 1 to 3.
- Follow same pattern as volume for load / save.
- Test with a Debug.Log from Level_01.

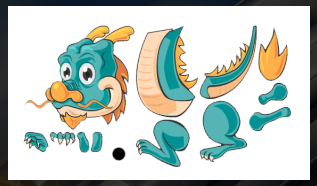
Sprite Sheet Animation

@UnityCourse
facebook.com/UnityCourse

In this video…

- The sprite sheet (AKA sprite atlas).
- Comparison to bone-based animation.
- Importing & slicing sprite sheets.
- Making your first animation.

Bone-Based vs Sprite Sheet / Atlas



Calculating Slice Grid

- Grid sheets usually have no padding.
- Grid width = image width / columns.
- Grid height = image height / rows.

Import Your Sprite Sheet

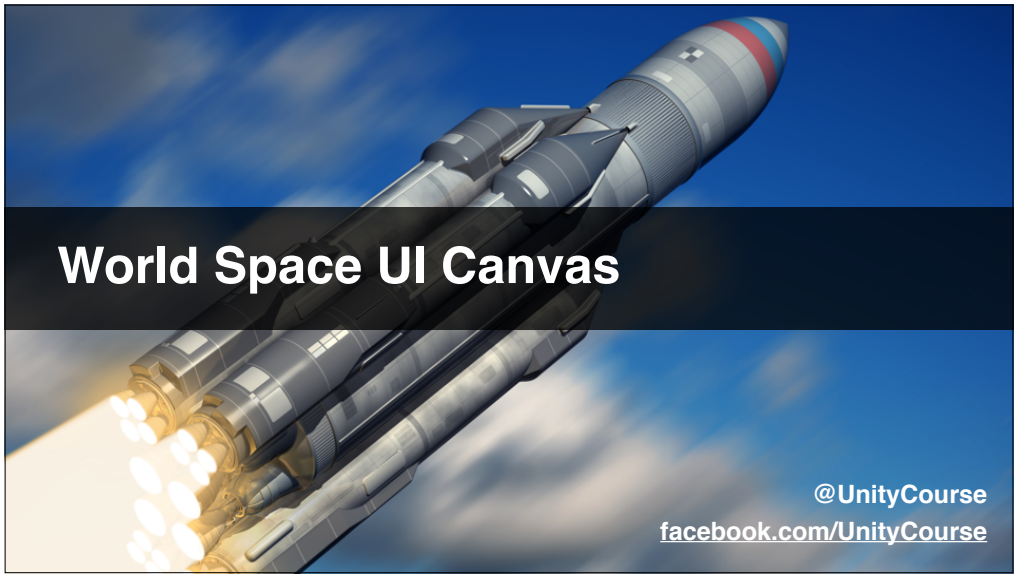
- Calculate the grid width / height.
- Slice you sheet and check it looks OK.

Making A 2D Animation

- Create an empty GameObject.
- Drag one of the sprites a placeholder.
- Add an **Animator** (animations being phased-out).
- Go into **Animation** window.
- [Create New Clip] from drop-down.

Make Your Animation

- Make and preview you own animation.
- Make sure the frame rate (“Samples”) is right.
- Don’t worry about it fitting in the scene for now.



In this video…

- Change to world space canvas for levels.
- Adjust grass tiling (using UV Rect).
- Add temporary “Core Game” panel.
- Translate & scale the level canvas.
- Adjust & prefab the camera.

About image aspect ratios

Adjust grass tiling (using UV Rect*)

- **12 grass squares wide in 16:9** aspect.
- In both aspects, that’s **6.75 grass squares tall.**
- There are **2 grass squares per tile.**
- Therefore we want W = 12 / 2 = **6 tiles wide overall.**
- Height therefore H = 6.75 / 2 = **3.375 tiles high.**

* *http://en.wikipedia.org/wiki/UV_mapping*

Add temporary “Core Game” panel

- Set to 1600 / 4 * 3 = 1200 pixels wide.
- Note it covers 9 columns as per GDD.
- Adjust to 5 rows high, leaves nice space
- Centre on the screen.
- This will be our word-space coordinate system.

Translate & scale the level canvas

- 1 grass square = 1m = 1 World Unit
- Width is 1600 pixels, and 12 World Units Wide
- “Pixels Per World Unit” = 1600 / 12 = 133.333…
- Scale Level Canvas 12 / 1600 = 0.0075 in X & Y

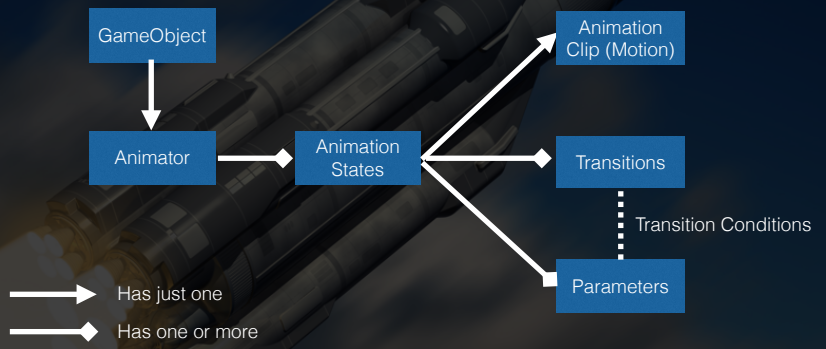
The Animation Controller

@UnityCourse
facebook.com/UnityCourse

In this video…

- How animators, states & motion clips relate
- Adding multiple animation states & clips.
- Options for transitioning between them.
- Again, only animate one character for now.

Animators, states & motion clips



In this video…

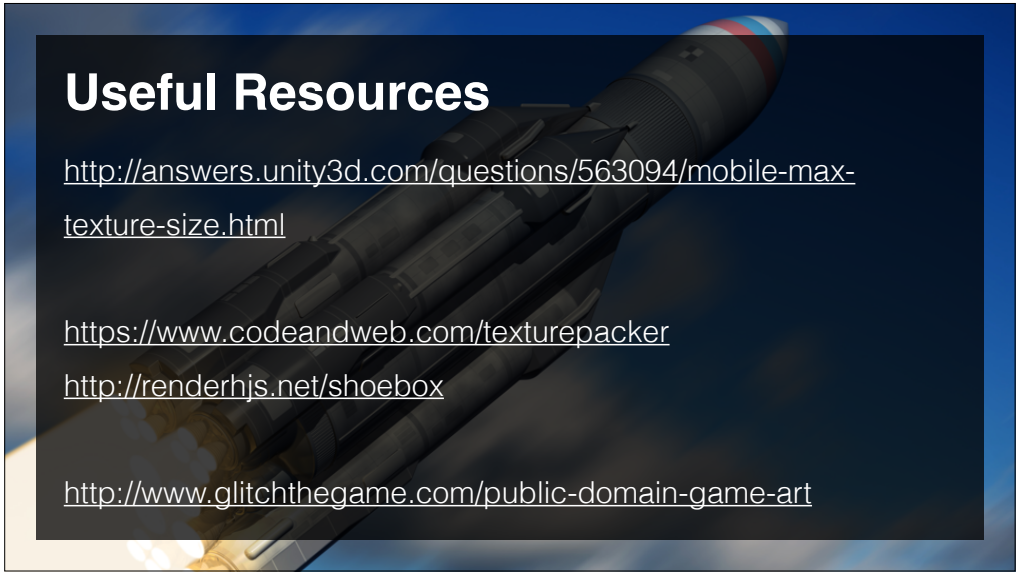
- Why my Lizard animation looked fuzzy.
- What to do about it.
- Max texture size for mobile devices.
- A bit about MIP Mapping while we're here.

MIP Mapping

- A technique where an original high-resolution texture map is scaled and filtered into multiple resolutions within the texture file.
- Takes up 33% more space, but MUCH faster.
- Not needed if we're at a fixed distance.

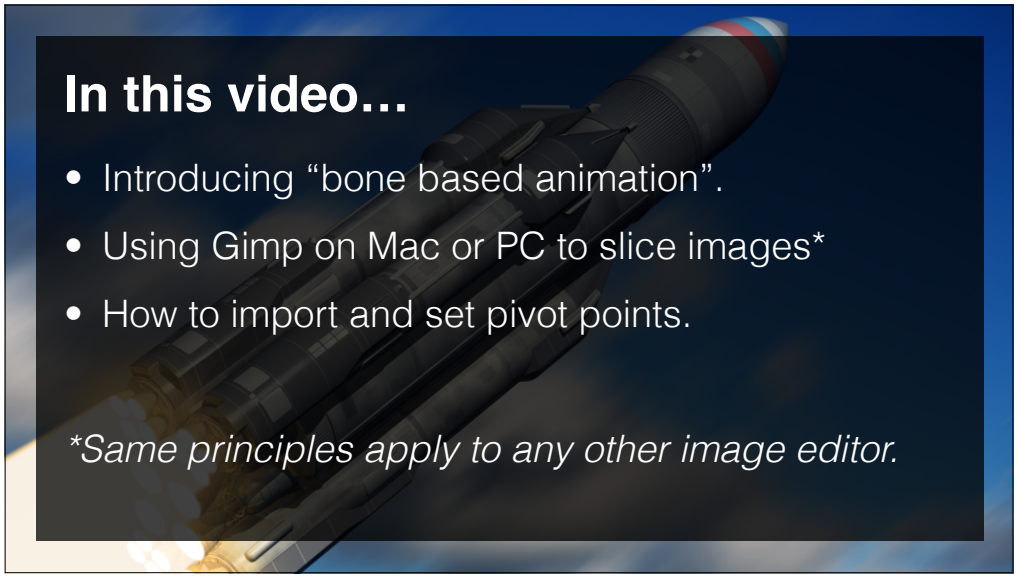
How to prevent fuzzy lizards

- Check your sprite sheet size in pixels.
- Look at the maximum x or y dimension.
- Ensure your texture "Max Size" is larger
- OR compact your sprite sheet.
- You can leave compression on though.


Useful Resources
http://answers.unity3d.com/questions/563094/mobile-max-texture-size.html
https://www.codeandweb.com/texturepacker
http://renderhjs.net/shoebox
http://www.glitchthegame.com/public-domain-game-art


SKIP TO "USING UNITY REMOTE"
@UnityCourse
facebook.com/UnityCourse


Using Gimp To Slice Images
@UnityCourse
facebook.com/UnityCourse


In this video…
• Introducing "bone based animation".
• Using Gimp on Mac or PC to slice images*
• How to import and set pivot points.
*Same principles apply to any other image editor.

Slicing images in Gimp

- **Image > Canvas Size…** increase by 2 pixels
- **Layer > Layer To Image Size** for yellow box
- **Tools > Selection Tools > Rectangle Select**
- Cut and paste section 2 pixels lower
- Save as PNG in Unity Assets folder

Prepare your sprite sheet

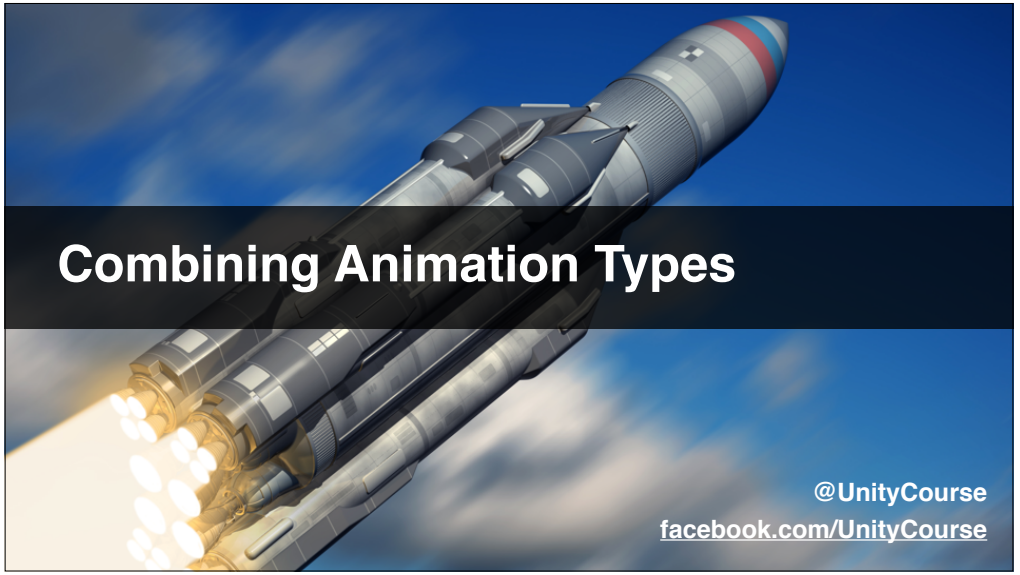
- Create at least one sprite sheet for bone animation.
- Make sure there's a pixel or two gap around limbs.
- Export with a transparent background.
- Import into Unity as multiple sprite.
- Set pivot points appropriately.

In this video…

- Animating Position, Rotation and Scale.
- Challenge: create your bone animation(s).

Create at least one bone animation

- Add life to a still sprite (e.g. gravestone or star).
- Move or rotate it's transform slightly.
- Make it loop this animation continuously.
- Give it at least two animation states.
- For example idle and attacking / attacked.



In this video…

- Different ways of animating objects.
- Different ways of moving transforms.
- Options for combining these.
- Challenge: You animate everything.

Different ways of moving objects

	Animation Types	Notes
Attackers	Sprite Sheet Script	Body animation by Sprite Sheet CG movement by Script
Defenders	Sprite Sheet & Bone N/A	Gnome by Sprite Sheet, Star Trophy Bone No defenders CG moves
Defender Projectiles	Bone-based Script	All using Position, Rotation & Scale CG moved by script

Animate position with animator XOR* script

Animate all your characters

- All attackers and defenders animated stationary.
- Use a mix of sprite and bone-based.
- Cycle through each state every 3 seconds.
- Put them all in a “showcase scene”.
- Share this scene on www.GameBucket.io

In this video…

- Giving our projectiles rotation in the animator.
- Giving them translation from the animator*
- Seeing the combined motion.

**We will change translation to script later.*

Animate your projectile rotation

- Get creative, you can scale too.
- Don’t spend too much time animating transform.
- Have fun, and share on www.GameBucket.io

Using Unity Remote

- What's Unity Remote and why's it useful.
- Unity Remote 4 on app stores (iOS and Android)
- How to use it.
- It's limitations.

http://docs.unity3d.com/Manual/UnityRemote4.html

Setting Up Unity

- Edit > Project Settings > Editor.
- Set "Unity Remote" Device to iOS or Android.
- Restart Unity if you're changing this.
- Set Game window resolution to that of device.
- Make the window at least this large if possible.

Setting Up Android

- Enable Developer options (see about 2 mins)…
  - Stay awake
  - USB debugging
  - Allow mock locations
- Settings > Storage > USB computer connection
  - Camera (PTP)

Setting Up iOS

- Allow connection on iOS and computer.
- Make sure the cable is plugged in!
- Try restarting app if it doesn't work.
- Also try restarting Unity.
- Click Play in editor each time you want to test.

In this video…

- Read music volume on load, improve Win & Lose.
- Catch 1st order error with **autoLoadLevelAfter().**
- Alternative fade without coding (thank Ryan).
- Save our scene of sprites & prefab everything.
- Our current project state is attached.

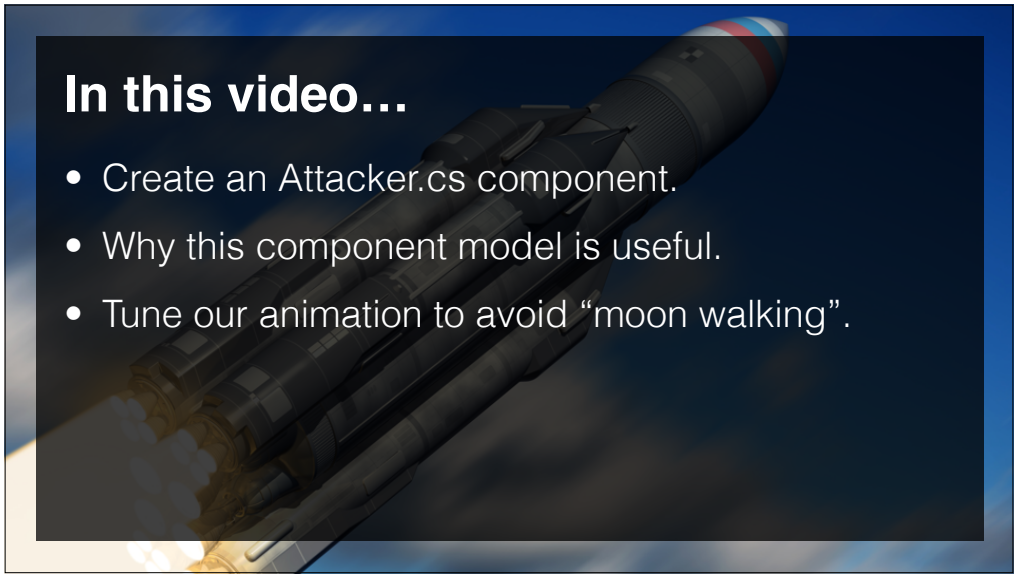
Set volume on Start scene

- Background music on Start should obey Options


?
Mid-Section QUIZ


ACT III: Scripting The Behaviour
@UnityCourse
facebook.com/UnityCourse


Moving Attackers From Script
@UnityCourse
facebook.com/UnityCourse


In this video…
• Create an Attacker.cs component.
• Why this component model is useful.
• Tune our animation to avoid “moon walking”.

Find & save walk speeds

- Find ideal speed for each attacker.
- Save these speed values back to prefabs.

Collision Matrix In Script

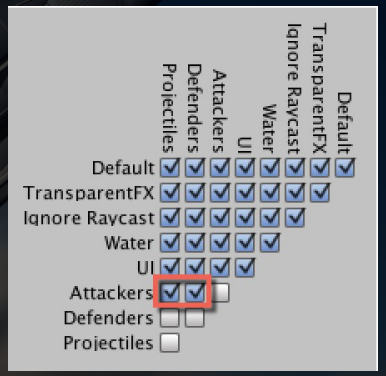
@UnityCourse
facebook.com/UnityCourse

In this video…

- Using **OnTriggerEnter2D (Collider2D collider).**
- Why we are using triggers not physics.
- Why we won't use the collision matrix this time.
- Adding appropriate colliders to all objects.

Why not use collision matrix this time

- Matrix would look like this.
- Must distinguish anyway.
- May as well keep all the decisions in one place.
- That place is the script.



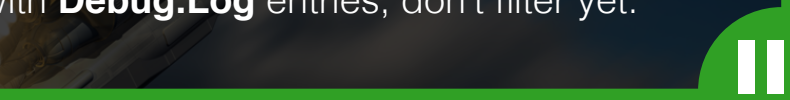
Collider interaction matrix (copy)

	Static Collider	Rigidbody Collider	Kinematic Rigidbody Collider	Static Trigger Collider	Rigidbody Trigger Collider	Kinematic Rigidbody Trigger Collider
Static Collider		collision			trigger	trigger
Rigidbody Collider	collision	collision	collision	trigger	trigger	trigger
Kinematic Rigidbody Collider		collision		trigger	trigger	trigger
Static Trigger Collider		trigger	trigger		trigger	trigger
Rigidbody Trigger Collider	trigger	trigger	trigger	trigger	trigger	trigger
Kinematic Rigidbody Trigger Collider	trigger	trigger	trigger	trigger	trigger	trigger

Derived from http://docs.unity3d.com/Manual/CollidersOverview.html

Add appropriate colliders to all

- Add 2D colliders (and RigidBody2D) on…
  - Attackers.
  - Defenders.
  - Projectiles.
- Test with **Debug.Log** entries, don't filter yet.



Using Animation Events

@UnityCourse
facebook.com/UnityCourse

In this video…

- The "what" and "why" of animation events.
- What methods can be called, and what can't.
- Modify Attacker.cs to accept speed events.
- Get animation transitions working for all attackers.
- Add "wishful" StrikeCurrentTarget() method.

Set walk speed at start of animation

- Add animation event.
- Select the appropriate function.
- Pass a speed parameter that stops feet slipping.
- Test it works in Play mode.

Add StrikeCurrentTarget() method

- Add **StrikeCurrentTarget (float damage)**
- Log something helpful for now.
- Add 1+ animation events to attackers.

In this video…

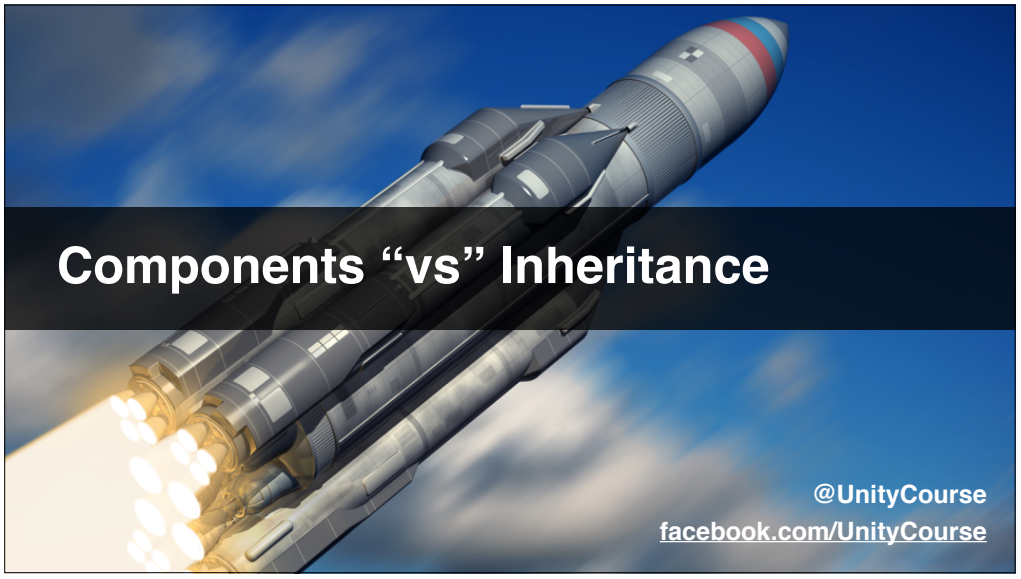
- Get our attacker animation transitions working.
- Temporary inspector access to attacker state.
- Ensure all animations transition smoothly.

Setup enum for attacker state

- Use appearing, attacking and walking.
- Ensure attackers only move left when walking.

Fix animations for other attackers

- Each one to appear in a lane of their own.
- Start with appear animation, or at X = 12.
- Walk left unless Attacking or Jumping.
- Keep walking once finished Attacking or Jumping.



In this video…

- The different approaches to abstraction.
- The benefits of a component model.
- Get **StrikeCurrentTarget()** working.

The difference

"Instead of sharing code between two classes by having them inherit from the same class, we do so by having them both own an instance of the same class."

http://gameprogrammingpatterns.com/component.html

Extendable behaviour…

SCRIPTS	Fox	Lizard	Mole	Jumping Mole
Stone	Jump	Attack	Burrow	Jump
Star Trophy	Attack	Attack	Attack	Attack
Cactus	Attack	Attack	Attack	Attack
Land Mine	Attack	Attack	Attack	Attack
Big Sone	Attack	Attack	Burrow	Burrow

Get StrikeCurrentTarget() working

- Add fox.cs and lizard.cs for special behaviour.
- Arrange for these scripts to talk to attacker.cs.
- They can also trigger the animator directly.
- Still no need to actually deal the damage yet.
- Hint: OnTriggerEnter2D on fox.cs and lizard.cs.

In this video…

- Why a separate component makes sense.
- Create & attach **Health.cs** component.
- Test destruction, and initial play tuning.

Create & attach Health.cs

- Create the script, with **public float health;**
- Attach to all defenders and attackers.
- Wire into Attacker.cs.
- Test in inspector that attackers deal damage.
- Bonus: Destroy defender when health == 0.

Animating Defenders & Projectiles

@UnityCourse
facebook.com/UnityCourse

In this video…

- Three approaches to 2D projectile animation.
- Separate defenders from their projectiles.
- Animate projectile using script **and** animator.
- Fix-up defender animation states.

Three different approaches

- Imagine you want a boomerang…
  1. Different animation for each start column.
  2. Create / modify animation from script.
  3. Do all translation from script.
- … we're going with option 3 for now.

Sort out your defender animations

- They sit in idle state unless **isAttacking** is set.
- When you tick isAttacking they fire continuously.
- Don't worry about detection of enemies in lane.
- May need separate attacking and firing states.
- Enjoy!

In this video…

- Why fire by animation events.
- Create **Shooter.cs** for shooting defenders.
- Create **FireGun()** method in Shooter class.
- Attach a gun gameObject to spawn projectiles.
- Arrange for animator to fire projectiles.

Get firing from animator working

- Create **Shooter.cs** for shooting defenders.
- Create **FireGun()** method in Shooter class.
- Create a gun gameObject to spawn projectiles.
- Arrange for animator to fire projectiles.

In this video…

- Why our Gnome fires too fast.
- Possible solutions to this type of issue.
- Why we choose to create a “fire” state.
- Fine-tune projectile size & spawn position.

Create fire state

- Create new state called “fire”.
- Transition into it after X loops of attack.
- Use attack motion in fire state for now.
- Move your Fire() event to fire state.
- Test it works.

Code for projectile damage

- Make projectiles damage **Attacker** with **Health**.
- Setup a play space, and start tuning.
- Tweak damage and health levels.
- We'll play tune again later.

In this video…

- Setting up buttons for defender (tower) selection.
- Initially they just toggle sprite colour.
- Setup **DefenderSelector.selectedDefender** static
- Test that static is set at start, and on button press.

Get sprite color toggling working

- Start with no defenders selected (all black)
- Clicking toggles to clicked defender.
- All unselected defenders blacked-out.
- Selected defender has different sprite colour.

Set selectedDefender static

- When a defender is selected, set the static.
- Test by logging from another method's **Update()**
- No need to place defenders yet.

Creating When Needed

@UnityCourse
facebook.com/UnityCourse

In this video…

- The problem with the Projectiles placeholder.
- Useful blog article on best practices*
- A pattern for checking and creating.

http://www.glenstevens.ca/unity3d-best-practices

Write code following this pattern...

```
myObject = FindMyObjectInScene();

if (myObjet == null)
{
    myObject = SpawnMyObject();
}
```

Tip #10 from here…

http://devmag.org.za/2012/07/12/50-tips-for-working-with-unity-best-practices

In this video…

- Ensure existing defenders' colliders mask square.
- Calculate the world-units position of a click.
- Calculate the nearest play-space grid centre.
- Spawn the currently selected defender there.

Calculating world-units of click

- **Input.mousePosition r**eturns pixel coordinates.
- **camera.ScreenToWorldPoint()** translates.
- Orthographic camera (no perspective)
- Distance from camera unimportant.

CalculateWorldPointOfMouseClick()

- Write the method.
- Test it works!

Vector2 SnapToGrid (Vector2 rawWorldPos)

- Write the method with this signature.
- Round to nearest whole-number world position.
- Hint: Consider **Mathf.RoundToInt ()**

In this video…

- Place enemy spawners.
- Decide how spawning is controlled.
- A word about the Flow Channel*
- Write script(s) to control spawning.

http://indiedevstories.com/2011/08/10/game-theory-applied-the-flow-channel

Options for controlling spawning

Method	Pros	Cons
Manually planned	Gives most design control.	Can be very time consuming. May miss interesting ideas.
Random / Procedural At design-time	Gives consistent experience. Can give best results.	Takes more time than run-time.
Random / Procedural At run-time	Very quick to setup. Can help re-playability.	Can lead to impossible play. Harder to play tune. Less consistent / “unfair”
Adaptive	Keeps the player in the flow.	Harder to implement.

Write Spawn() method

- void Spawn (GameObject myGameObject).
- Remember to parent to this spawner.
- Test it works.

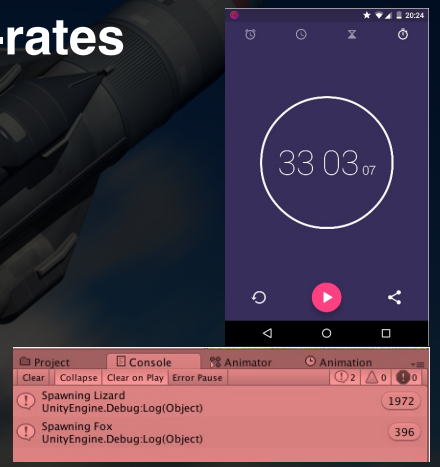
Write isTimeToSpawn()

- bool isTimeToSpawn (GameObject attacker).
- Return true to spawn this frame.
- Return false if not.
- Bonus: log warning based on “frame-rate cap”.

Check your spawn-rates

- Total seconds = 1983
- Lizard rate = 1.01 (3sf)
- Fox rate = 5.01 (3st)

Pretty good, within 1%



In this video…

- Find a way of defenders detecting attackers.
- Only shoot at attackers if ahead in lane.
- Modify **Shooter.cs** to make this work.
- Test that defender enter and leave “attack” state.

Write code to find lane spawner

- **void setMyLaneSpawner ()**
- Looks through all the **Spawners**
- Finds the with your y-position
- Sets the **private Spawner myLaneSpawner**
- Returns an error if you can’t

Write attacker detection code

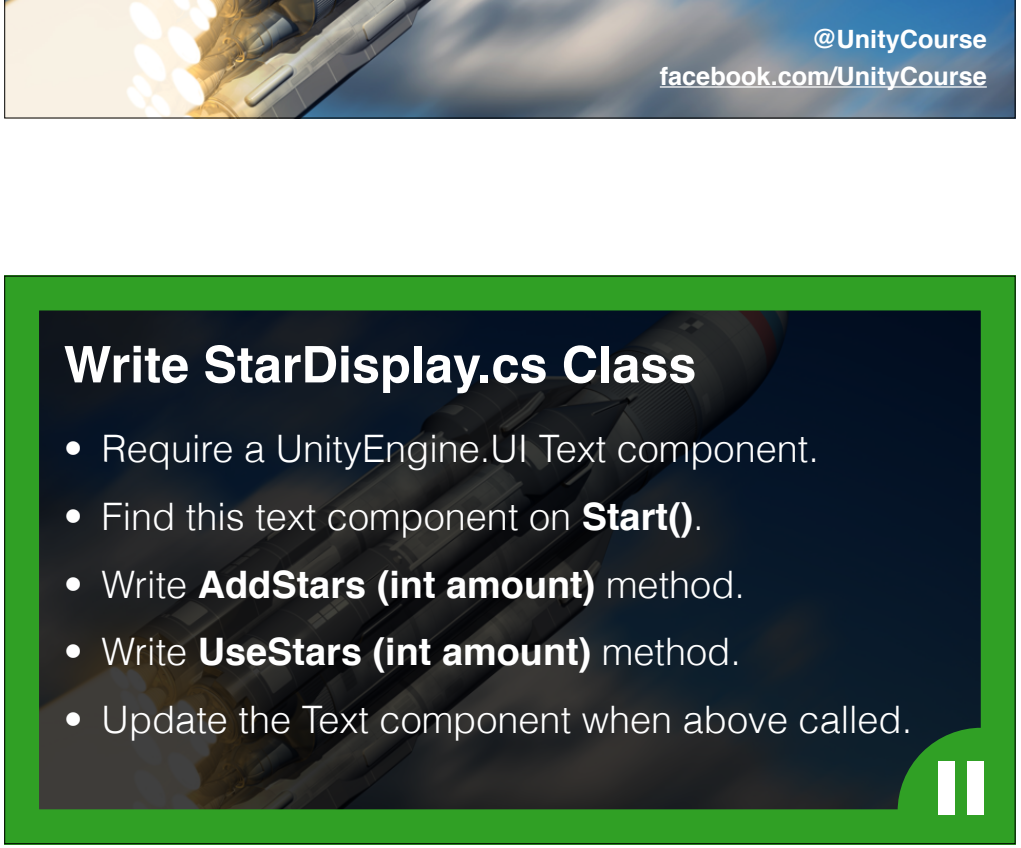
- Signature: **bool isAttackerInLane ()**
- Check they enter attack state.
- Check they leave when no enemies.

In this video…

- Add a sun scoreboard to the game space.
- Star Trophy animation calls script to add sun.
- Write **StarDisplay.cs** class to update scoreboard.
- Write **defender.AddStars(int amount)** method.
- Wire these scripts together.

Write StarDisplay.cs Class

- Require a UnityEngine.UI Text component.
- Find this text component on **Start()**.
- Write **AddStars (int amount)** method.
- Write **UseStars (int amount)** method.
- Update the Text component when above called.



In this video…

- Assign a star cost to every defender.
- Prevent placement until you can afford it.
- Spend stars when defenders are placed.
- Use an **enum**eration to pass meaning.
- Rough play tuning to create a challenge.

Update DefenderSpawner.cs

- Spawn and use stars if can afford.
- Otherwise **Debug.Log ("Insufficient stars")**
- Test this works properly.

In this video…

- Remove lose test button.
- Create a lose collider.
- Setup lose triggering & transition.
- Improve lose screen.

Write LoseCollider.cs Class

- Find the LevelManager.
- Use OnTriggerEnter2D to detect loss.
- Call **LevelManager.LoadLevel ("03b Lose");**

In this video…

- Create a UI slider to visually show level progress.
- Make the slider to "count down" to level end.
- When time runs out…
  - Show "You Survived" text, and play a sound.
  - Auto-load next level.

Write GameTimer.cs Class

- Manage the level time (start and remaining).
- Move the slider every frame.
- Play music, display message and load level.

In this video…

- Tidy **Spawner.cs > isTimeToSpawn()**
- Adjust colliders so attackers hit defenders.
- Fix the gravestone animation transitions.
- Creates prefabs of our work.

Write Stone.cs script

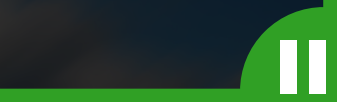
- Attacker colliding sets **underAttack trigger.**
- Ensure that projectiles don't trigger.
- Check animation transitions happen as expected.
- Hint: Use **OnTriggerStay2D** ()

In this video…

- Display the defender cost on buttons.
- Tweak the spawn frequency of attackers.
- Adjust the health of attackers & defenders.
- Choose amount of damage for projectiles.
- Play and make sure it's a challenge.

Modify button.cs for defender cost

- Read **starCost** from relevant defender.
- Display the cost on the button in small text.
- Log a warning if you can't find the cost text.



Play tune your game

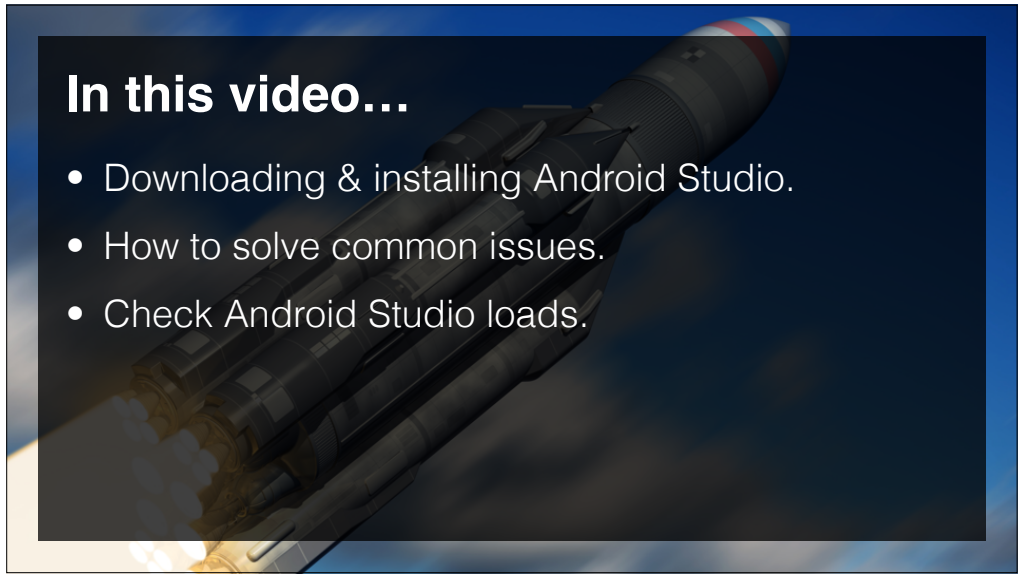
- Choose parameters that create challenge.
- Make each level introduce a new game mechanic.
- Also make each level a little harder.

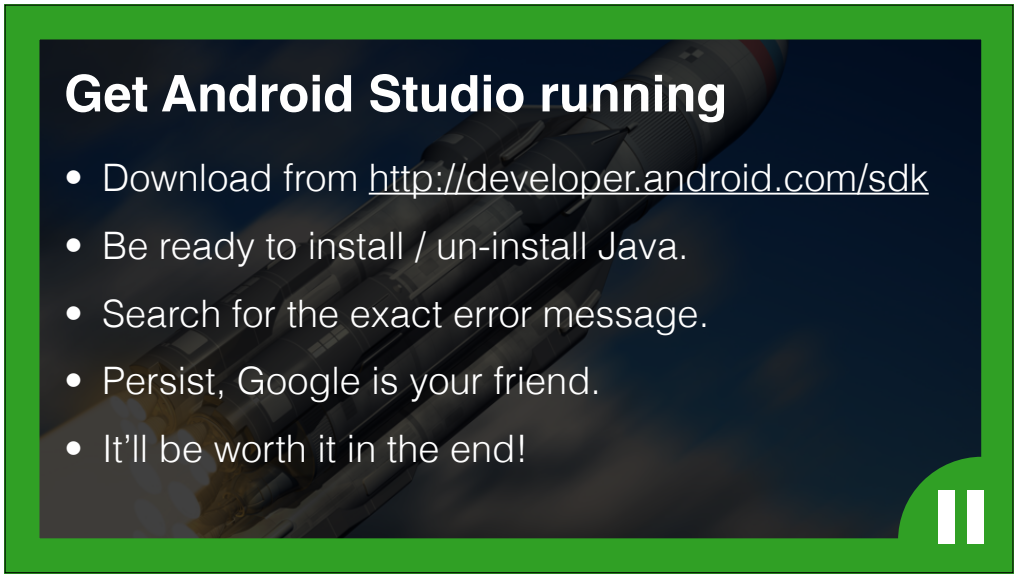
?

End of Section QUIZ


ACT IV: Mobile Deployment
@UnityCourse
facebook.com/UnityCourse


Installing Android Studio
@UnityCourse
facebook.com/UnityCourse


In this video…
Downloading & installing Android Studio.
How to solve common issues.
Check Android Studio loads.


Get Android Studio running
Download from http://developer.android.com/sdk
Be ready to install / un-install Java.
Search for the exact error message.
Persist, Google is your friend.
It'll be worth it in the end!

In this video…

- Setting up Build Settings for Android.
- Deploying to Android device*
- Play testing on the device.

** You'll need a device connected with a USB cable.*

Deploy to Android and share

- Get it running on your device.
- Excitedly show somebody… anybody!
- Capture a screenshot.
- Share in the discussions / Facebook group*

** https://www.facebook.com/groups/completeunitydeveloper*

In this video…

- Setup Build Settings in Unity.
- Build to iOS simulator (Mac “needed”).
- To build to physical device you “need” a dev kit.
- Briefly play-test, and note improvements.
- Share your creation with the world.

Deploy to iOS and share

- Get it running on your iOS simulator / device.
- Excitedly show somebody… anybody!
- Capture a screenshot.
- Share in the discussions / Facebook group*

* *https://www.facebook.com/groups/completeunitydeveloper*

User Testing Tweaks

@UnityCourse
facebook.com/UnityCourse

In this video…

- Simplify by removing **SetStartVolume.cs***
- Destroy tagged game objects on Win condition.
- … this also solves the “You Win” issue.
- Add a simple STOP button to game.

* *Thanks to Marko for suggesting this.*

Write DestroyAllTaggedObjects()

- Finds all objects with **destroyOnWin** tag.
- Be very careful with tag spelling.
- Test by setting all but one defender type.
- Tag all defenders, attackers & projectile prefabs.
- Test by playing to end of a level.

Add a simple STOP button

- Somewhere along the bottom maybe.
- Take them back to the Start scene.
- Extra credit: Pop-up allowing continue.

In this video…

- Testing after upgrading to Unity 5.
- Fixing Quit button opacity / depth.
- Try new Rigidbody constraints