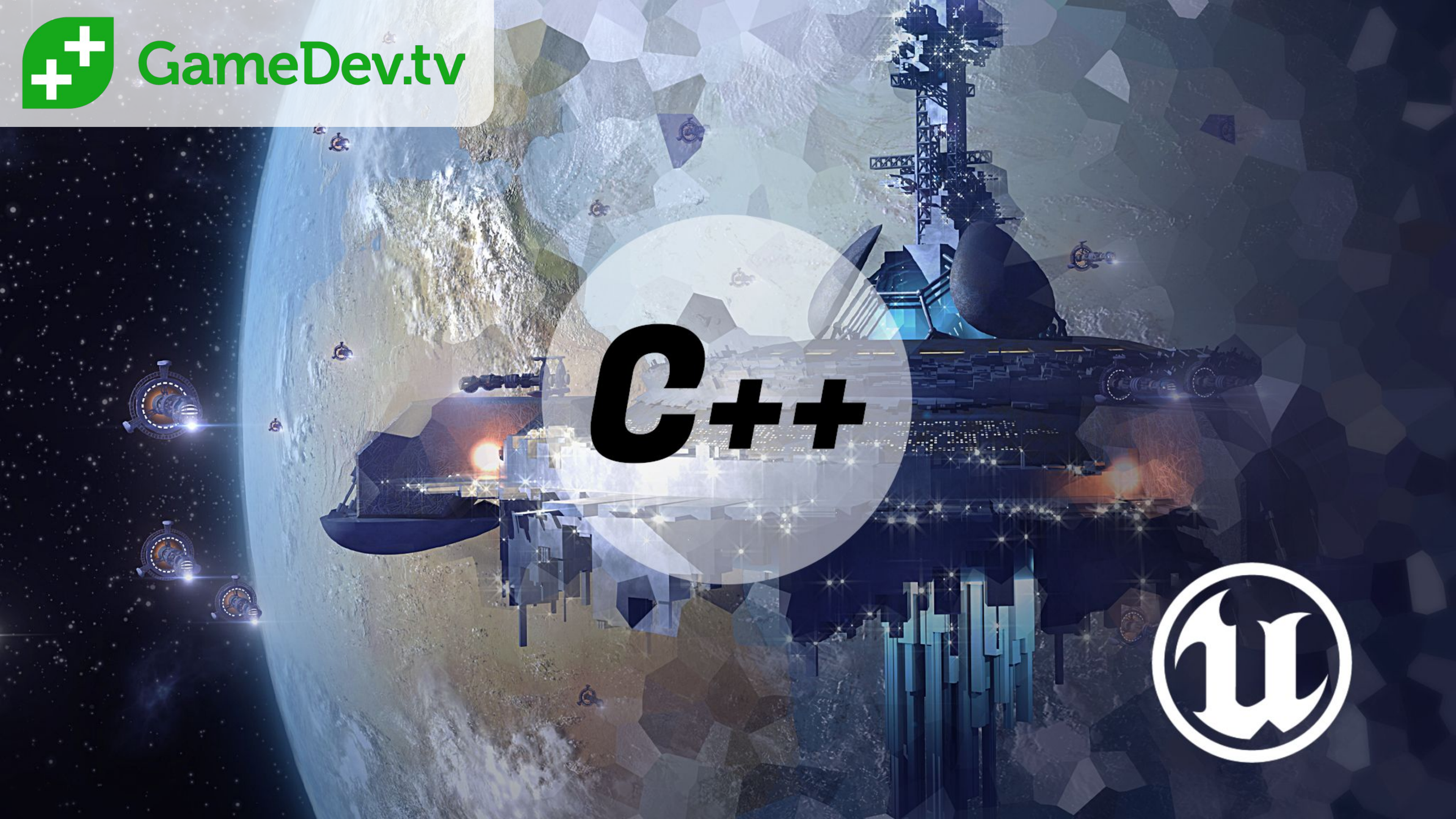

GameDev.tv
C++

GameDev.tv

Section 4: Building Escape

gdev.tv/uc2slides

GameDev.tv

What's Coming Up

gdev.tv/uc2slides

GameDev.tv

Introduction To Building Escape

gdev.tv/uc2slides

The Building Escape Project

- Understand the game we are going to create.
- Setting Up a new project.
- Interacting with a physical world.
- Level design, lighting and volumetrics.
- Further develop our Unreal C++.

Open Bull Cow Game

- Make sure you are using Unreal Engine 4.22
- Download the BullCowGame-starter-kit.zip
- Extract the Files
- Check you can launch the game.

GameDev.tv

Building Escape End Goal

gdev.tv/uc2slides

GameDev.tv

Setting Up The Building Escape Project

gdev.tv/uc2slides

The Building Escape Project

- Setting up a brand new project.
- Show how to delete old or test projects.

Setup Your New C++ Project

- Create a new C++ project. With or without the starter content.
- If using version control check out the .gitignore included with this lecture.
- Play about in your level, get comfortable.

GameDev.tv

Pointer Primer

gdev.tv/uc2slides

Pointers

- When you see/use * next to a type.
- Pointers are memory address.
- Like references they save having to copy / move data in memory.
- You can point to any object.
- You can lose control of your data.

Pointer Syntax

- `FType* NameOfPointer`, `FType * NameOfPointer`, `FType *NameOfPointer`
- All three statements are equivalent, we use 1st
- In all cases `NameOfPointer` is a pointer.
- In all cases the type of the object pointed to is `FType`.

Accessing Members

- Imagine we have `AActor* SomeActor;`
- The `AActor` class has a method `GetName()`.
- `*SomeActor` “de-references” the pointer.
- You could write `(*SomeActor).GetName();`
- But you can follow and access in one using `->`.
- We access name with `SomeActor->GetName()`.

GameDev.tv

Unreal's Classes and Components

gdev.tv/uc2slides

Classes and Components

- Introducing the idea of inheritance.
- Explore using the Class Viewer to see the depth of Unreal’s class system.
- Inheritance for “is a” relationships.
- Components for “has a” relationships.

Classes and Inheritance

- e.g. Character “is a” Pawn, Pawn “is an” Actor.
- c.f. Dog “is a” Mammal, Mammal “is an” Animal.
- Unreal makes extensive use of inheritance.
- Is a powerful tool if used properly.
- Can be inflexible and hard to re-factor.

Components

- Components are great for sharing a common behaviour or features.
- Some components are necessary- mesh, collision, audio can only be components.
- Actors can have a custom components...
- Let’s create WorldPosition component...

Challenge

- Create another actor in your scene.
- Add to it your newly made component.

GameDev.tv

Deleting A Class

gdev.tv/uc2slides

GameDev.tv

Logging To The Output Log

gdev.tv/uc2slides

Run-Time Messages For Feedback.

- Using the Unreal output log.
- `UE_LOG(Category, Verbosity, TEXT("Message"));`
- `UE_LOG(LogTemp, Warning, TEXT("Hello!"));`
- Error = Red.
- Warning = Yellow.
- Display = Grey.

Challenge

- Create a log message that is red and grey as well.
- You’ll end up with 3 messages in total per component.

GameDev.tv

Project Settings: Default Startup Level

gdev.tv/uc2slides

GameDev.tv

Accessing An Object's Name

gdev.tv/uc2slides

Using Pointers

- We are going to play with pointers.
- Learn about operator precedence.
- Use `GetOwner()` to find the components owner.
- `AActor*` is a pointer to an actor.
- Use `->` to access members.
- Use `GetName()` to find the objects name.
- Exposed to IWYU.

Challenge

- Create a variable to store the objects name.
- Output the name of the object to the console in your log.

GameDev.tv

Getting An Actor's Transform

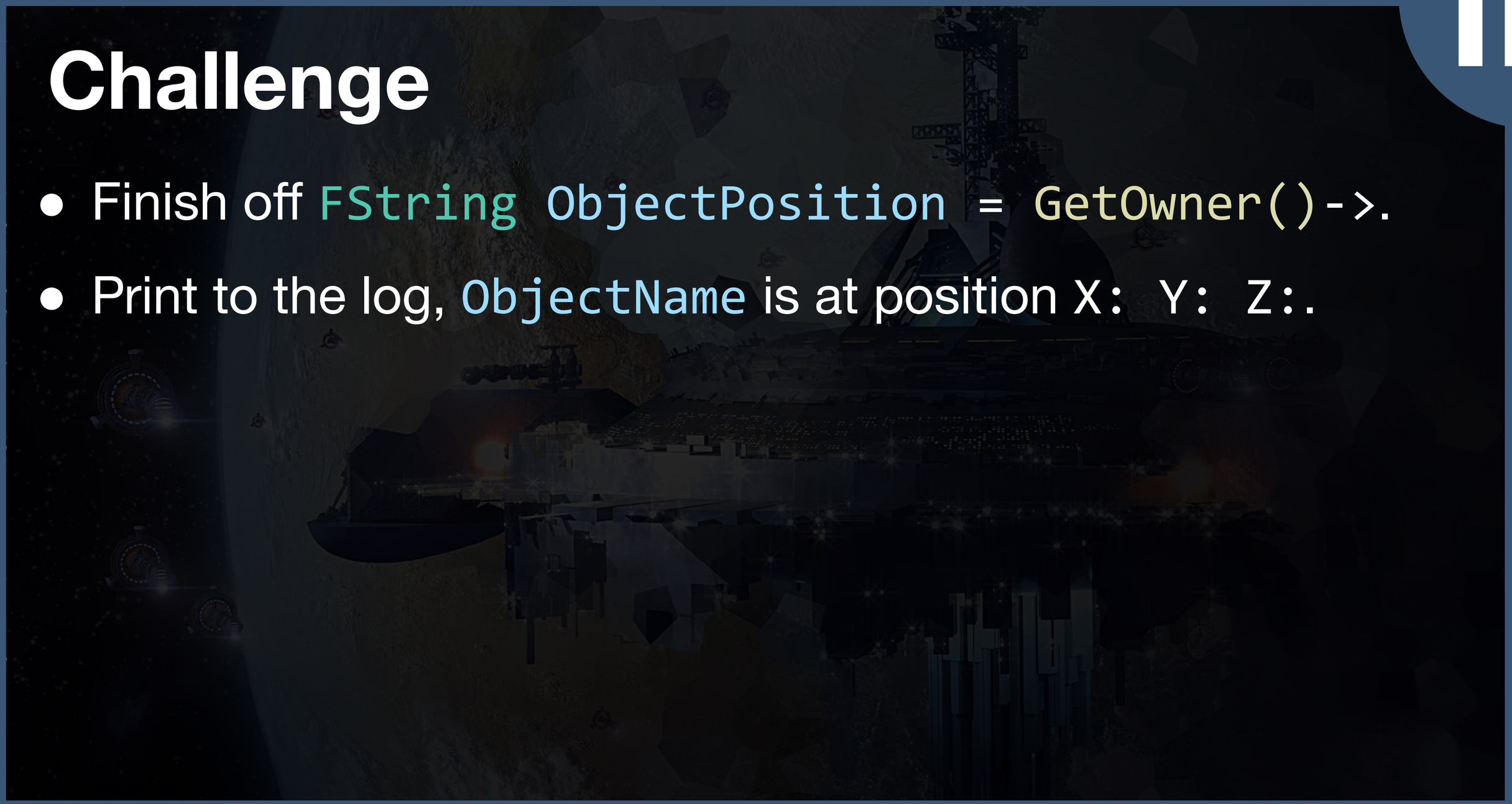
gdev.tv/uc2slides

Getting The Location

- Finish the `WorldPostition` component.
- Using `.` and `->` to access members.
- Introduce `FVector`.

Challenge

- Finish off `FString ObjectPosition = GetOwner()->`.
- Print to the log, `ObjectName` is at position `X: Y: Z:`.



GameDev.tv

Importing Custom Meshes

gdev.tv/uc2slides

FBX Importing.

- We need something in our scene to move.
- Importing a mesh and texture.
- Assigning a material and texture to our model.
- Getting more textures.

Get The Door To Open

- Create a an `FRotator` with a Yaw Value of 90/-90 degrees.
- Pass it into `SetActorRotation()`.

GameDev.tv

Using BSP For Basic Building Blocks

gdev.tv/uc2slides

Binary Space Partitioning

- Quicking mocking up an idea or level design.
- Look at the various brushes.
- Order matters. - It can be changed.
- Rebuilding helps solve issues with BSP.
- Work aligned to grid.

Create Your Test Level

- Create a room.
- Walls must be 3m high.
- Remember to organise your scene.
- Create a doorway that is 2m in height and 1m in width on one side of the room.
- Place your door in that hole.

GameDev.tv

BSP Challenge

gdev.tv/uc2slides

GameDev.tv

Basic Lighting

gdev.tv/uc2slides

GameDev.tv

Transforming Material Textures

gdev.tv/uc2slides

Scaling Textures

- Create a basic material.
- Setup some parameters to scale the texture.
- Show you how to scale the texture horizontally and vertically independently.

Texture Your Test Level

- Texture the rest of your test level.
- Use only the one texture and scale it appropriately for the object you are applying it to.

GameDev.tv

Rotating An Actor With Code

gdev.tv/uc2slides

Opening The Door

- Use code to open our door.
- Another example of structs.
- Introduce the struct `FRotator`.
- Use `floats` for the first time.

Texture Your Test Level

- Texture the rest of your test level.
- Use only the one texture and scale it appropriately for the object you are applying it to.

GameDev.tv

Object Collision

gdev.tv/uc2slides

We Are Stuck!

- There are 4 Options:
- Remove the collider.
- Use complex collision.
- Use BSP to static mesh.
- Get the artist to provide assets with collision.

GameDev.tv

Using Linear Interpolation

gdev.tv/uc2slides

Linear Interpolation

- `FMath::Lerp(Start, End, %Distance)`.
- Start: 0, End 10, % of 10% (or 0.1).

Frame	Returned Value	End	10% of Remaining Distance
0	0	10	1
1	1.00	10	0.9
2	1.90	10	0.81
3	2.71	10	0.73
4	3.44	10	0.66
5	4.10	10	0.59
6	4.69	10	0.53

Linear Interpolation

- We only have to change the Yaw value, not all three.
- https://en.wikipedia.org/wiki/Linear_interpolation

Open The Door...

- Use a Lerp to open the door.
- You'll need to update the starting yaw value.
- This code will only have an effect if your door has a yaw value that isn't 90. We solve this in the next lecture…
- Extra credit if you know how to solve it now!

GameDev.tv

Relative Vs Absolute

gdev.tv/uc2slides

Open The Door Anywhere...

- Easy to get caught out.
- In `BeginPlay()` you will need to set the initial values for the following member variables: `InitialYaw`, `CurrentYaw` and `TargetYaw`.
- Remember to step through your code step by step.

GameDev.tv

Assets Naming Convention

gdev.tv/uc2slides

GameDev.tv

Exposing Parameters To The Editor

gdev.tv/uc2slides

We Are Stuck!

- There are 4 Options:
- Remove the collider.
- Use complex collision.
- Use BSP to static mesh.
- Get the artist to provide assets with collision.

Open The Door Anywhere...

- Texture the rest of your test level.
- Use only the one texture and scale it appropriately for the object you are applying it to.

GameDev.tv

Framerate Independent Using DeltaTime

gdev.tv/uc2slides

Variable Door Speed

- The doors should open up at a consistent speed.
- Currently they will open up faster on faster computers.
- DeltaTime is the time between frames being rendered.
- 60fps = 1/60th second, 0.01666s or 16.66ms.

GameDev.tv

Trigger Volumes

gdev.tv/uc2slides

Trigger Volumes

- Add a trigger volume to our scene.
- Jump straight into a challenge.

Assign The Trigger Volume

- Declare `ATriggerVolume* PressurePlate;`.
- You'll need the right #include.
- Make the `PressurePlate` a `UPROPERTY(EditAnywhere)`.
- In the UE4 Editior on the OpenDoor component, set the trigger volume to the one we have placed in our scene.
- Anything with the OpenDoor component but not set which volume will crash UE4 upon play... when we implement it.

GameDev.tv

Using Collision Volumes

gdev.tv/uc2slides

Trigger Volumes

- Trigger volumes are very useful tools.
- Volume that can detect things entering / leaving.
- We are using one as a pressure plate.
- How we can choose what can open our door.
- Use `IsOverlappingActor()` on `ATriggerVolume`.
- Polling vs using events.

Time To Refactor

- Put the code we have created in `TickComponent()` into a new function `OpenDoor()`.
- Remember you'll need to add code into the header file.
- Then we will finish off our code.

GameDev.tv

Protecting From A Null Pointer

gdev.tv/uc2slides

GameDev.tv

Getting The Player To Open The Door

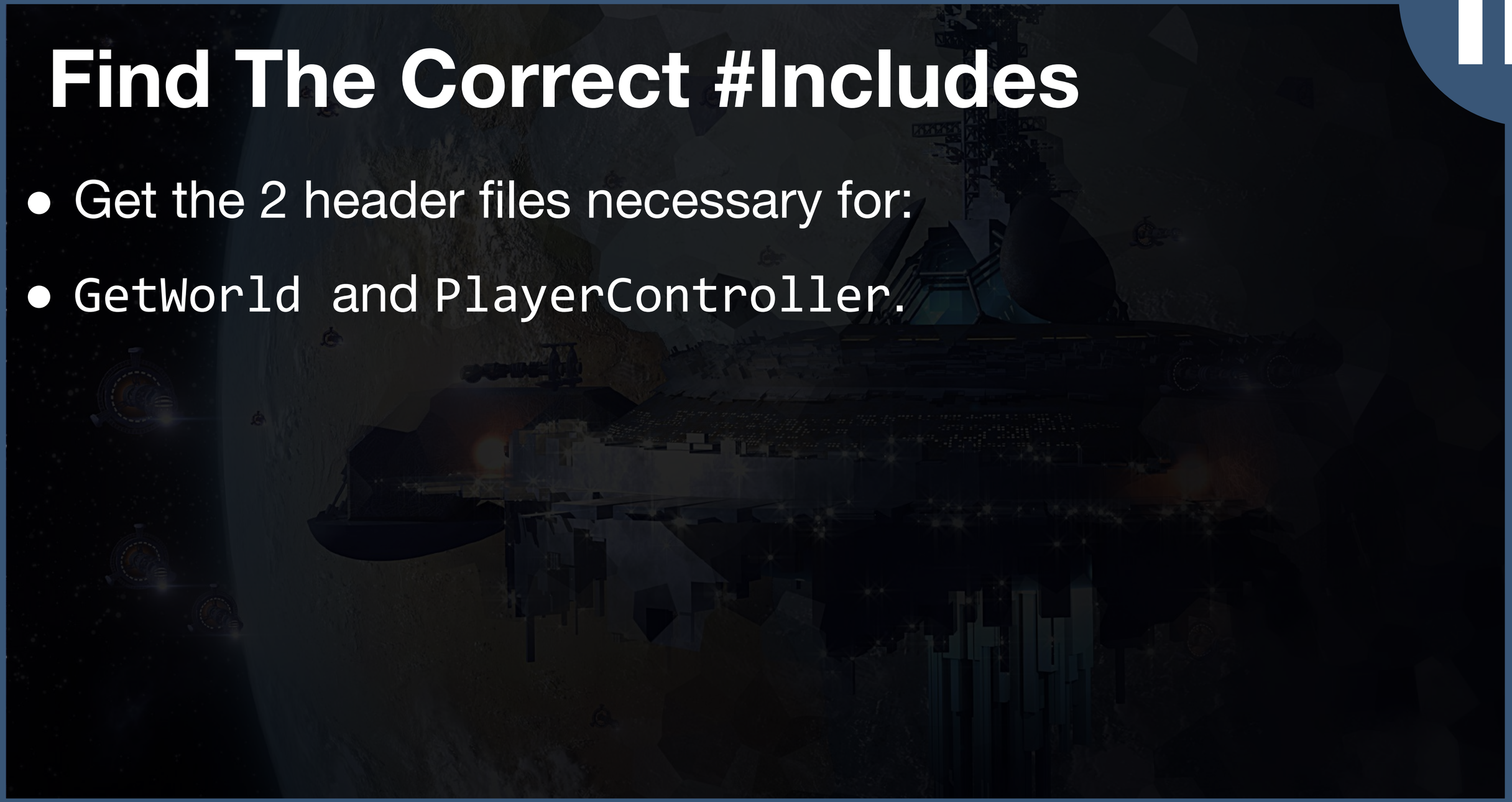
gdev.tv/uc2slides

Player Interaction.

- We've used `GetOwner()` to search "bottom-up"
- Now let's use `GetWorld()` to search "top-down"
- Game Mode specifies the Default Pawn Class.
- The Default Pawn is your "body", is transient.
- The Player Controller is your "mind", persist.
- PlayerController class has `GetPawn()`.

Find The Correct #Includes

- Get the 2 header files necessary for:
- `GetWorld` and `PlayerController`.



GameDev.tv

Getting The Door To Close

gdev.tv/uc2slides

Close The Door(s)

- Create the necessary code to close the door.
- The door(s) should be closed whenever the player is not in the trigger volume.

GameDev.tv

Using GetTimeSeconds()

gdev.tv/uc2slides

Play Tuning Our Game

- Using `GetWorld()->GetTimeSeconds()`.
- Making our game highly “play tunable”.
- Using a spotlight to provide “affordance”.
- Play-testing to ensure the game is annoying!

Delay The Doors Closing

- Write some simple timing code.
- Get the doors closing after a 2s delay.
- Play-test to ensure you can’t escape.

GameDev.tv

Designer Friendly Components

gdev.tv/uc2slides

Designer Friendly

- Remove magic numbers from the code.
- Think about what parameters you want to expose.
- Expose them to the editor.

GameDev.tv

Grabbing System Overview

gdev.tv/uc2slides

Overview

- Outline the end result.
- You try and think how it may be done.
- I’ll outline how we’ll be doing it.

Write Down Your Ideas

- Use the knowledge you have already.
- Would you use a component or inheritance?
- Hint: either could work, just hear yourself reason.
- How may you know what to grab?
- What game object would you be working with?
- Share your ideas for discussion.

The Grabbing System Overview

- We want to be able to lift the cone next.
- We'll add a `Grabber` component to the player.
- The player is a temporary actor, appears on play.
- The Game Mode sets which Default Pawn to use.
- Create Default Pawn & Game Mode Blueprints.
- Specify our modified Default Pawn.

About GameMode

- Anything from:
  - What inventory items a player starts with.
  - How many lives are available.
  - Time limits.
  - Score needed to end the game.
- These all belong to GameMode.

https://docs.unrealengine.com/latest/INT/Gameplay/Framework/GameMode/index.html

GameDev.tv

Modifying The Default Pawn Actor

gdev.tv/uc2slides

The Default Pawn

- Why Blueprint is helpful in this case.
- How to make a Blueprint from the Default Pawn.
- Note this Blueprint class inherits, an “is a” relation.
- A Blueprint is like a template.
- You make an “instance” in the scene.
- Explore “instantiating” from Blueprint & modifying.

Try Making A Rugby Ball Pawn!

- Modify the DefaultPawn_BP somehow...
- ...scaling on one axis for example.
- Create an instance by dragging into the world.
- See how modifying instance doesn’t change BP.
- Revert your change.

GameDev.tv

Inherit Game Mode Blueprint

gdev.tv/uc2slides

...From a GameMode

- “Hard coding” means assets written into code
- The `DefaultPawn_BP` is an asset.
- We want to be able to track changes to its name.
- It is convenient to use Blueprint for this purpose.
- Extending our C++ Game Mode with Blueprint.
- Selecting the new `DefaultPawn_BP`.
-

Blueprint The GameMode

- Find the C++ Game Mode in the Content Browser.
- Create a Blueprint class derived (inheriting) from it.
- Set this as the Default GameMode in…
- Settings > Project Settings > Maps & Modes.
- Make sure the game still runs the same.

GameDev.tv

Getting The Players Viewpoint

gdev.tv/uc2slides

Where Are We Looking?

- Know where the player is looking.
- Another look at out-parameters.
- A way of marking-up out parameters.
- Continuously logging player viewpoint.

Log the Viewpoint Every Tick

- Log the viewpoint position and direction every tick.
- Hint: You may need to use `ToString()`.
- Get used to working with different data types.
- Give it at least 20 mins if you’re struggling.
- Carry on watching for my solution.

GameDev.tv

Using DrawDebugLine

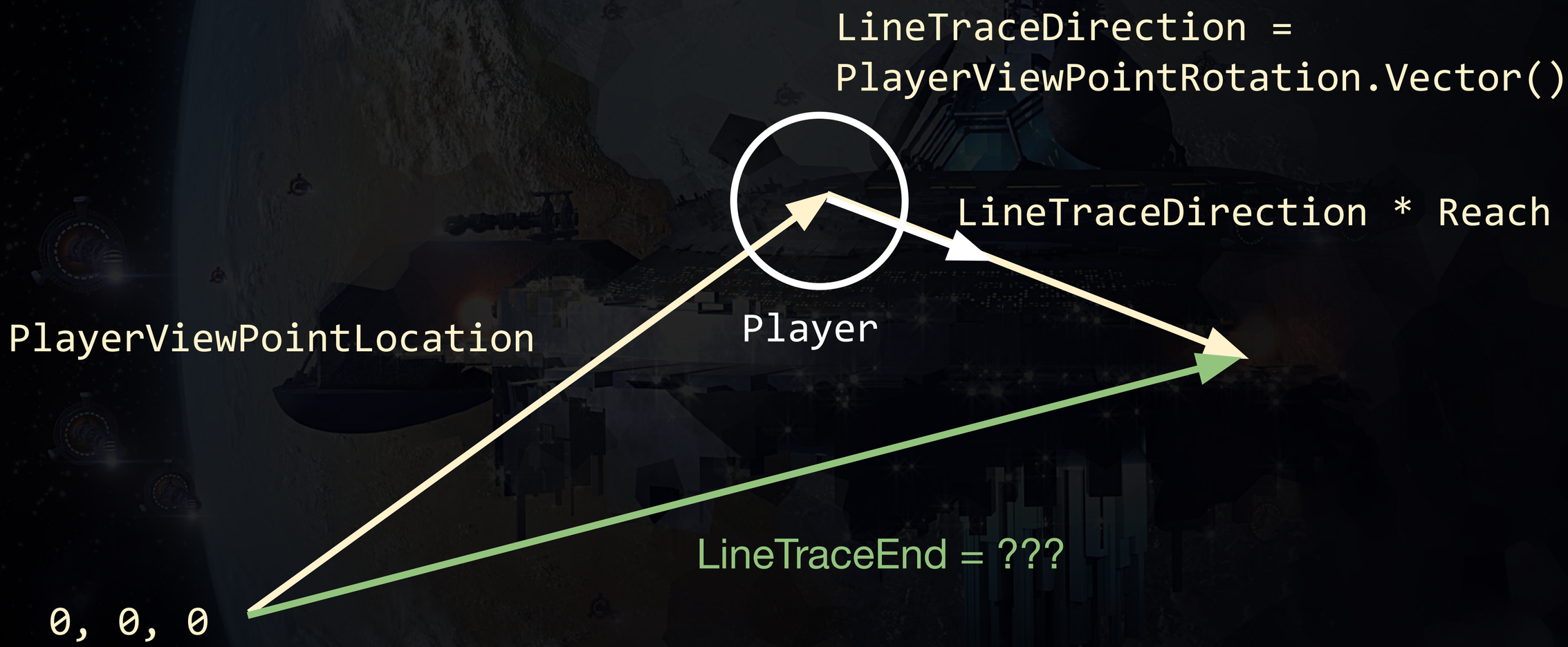
gdev.tv/uc2slides

Where Are We Looking?

- How to add vectors.
- Calculating our line trace end point.
- Using debug functions for visualisation in Unreal.
- Use `DrawDebugLine()` to visualise the vectors.

Adding Vectors

Calculating `LineTraceEnd`



Calculate `LineTraceEnd`

- Create a private variable `float Reach = 100.f;`.
- Calculate `LineTraceEnd`.
- Test the debug trace, eject to visualise (F8).

GameDev.tv

Line Tracing AKA Ray-Casting

gdev.tv/uc2slides

Ray Casting

- Line tracing (AKA ray casting) is a very useful tool.
- Imagine we shine a virtual laser into the world.
- We can use different view modes to visualise.
- Simulating physics sets the object channel.

Calculate `LineTraceEnd`

- Create a private variable `float Reach = 100.f;`.
- Calculate `LineTraceEnd`.
- Test the debug trace, eject to visualise (F8).

GameDev.tv

Using FindComponentByClass()

gdev.tv/uc2slides

In This Video...

- What `FindComponentByClass()` does.
- How to use it to find attached components.
- Introducing angle brackets <> for generics.
- Use nullptr to initialise your pointers.
- Log a useful error if the component isn’t attached.

Log An Error.

- Log at `Error` verbosity if no component found.
- Write an error that helps the reader fix the issue.
- Find and include the name of the object.
- … in this case it’s the Default Pawn.
- Temporarily remove component to test.

GameDev.tv

Introducing Input Binding

gdev.tv/uc2slides

In This Video...

- Settings > Project Settings > Engine > Input.
- Action mappings are used for on / off actions.
- Axis mappings are used for analog values.
- You can give players a way or re-mapping.
- Many keys can bind to one action.
- How to call a function on a key press or release.

Find The Input Component

- Create an appropriate private member.
- Check for the component as Physics Handle.
- Log a similarly helpful error if it’s not attached.
- Don’t bother trying to remove to test this time.

GameDev.tv

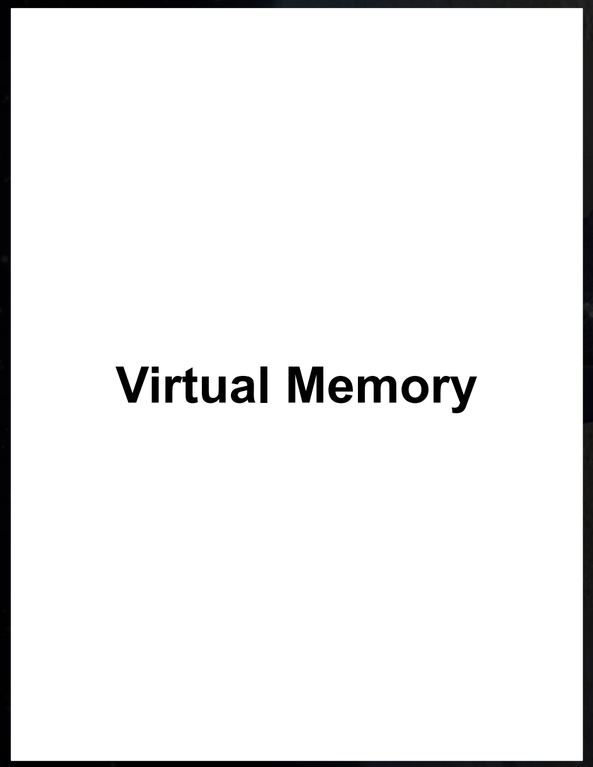
Accessors & Memory Layout

gdev.tv/uc2slides

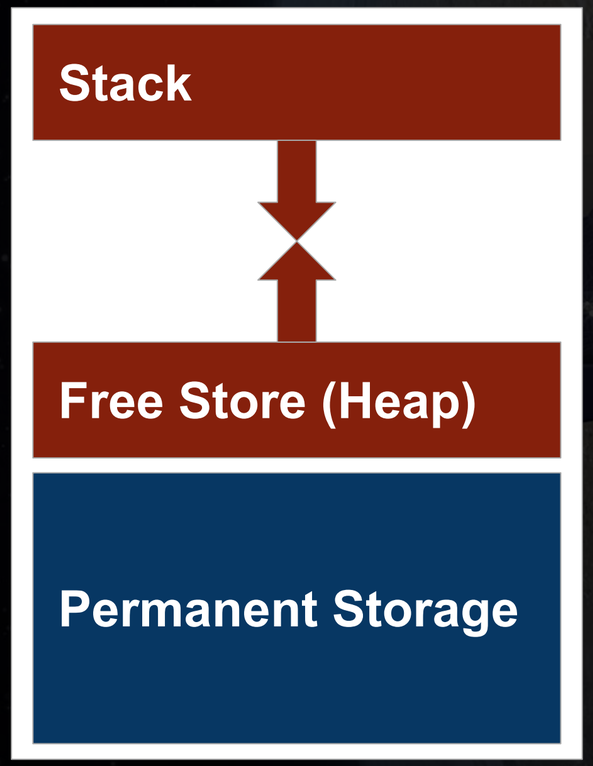
In This Video...

- How the arrow, dot and :: accessors work.
- Introducing virtual memory.
- Introducing permanent storage, stack & heap.
- Heap is also known as free store.
- How accessor operators relate to memory.

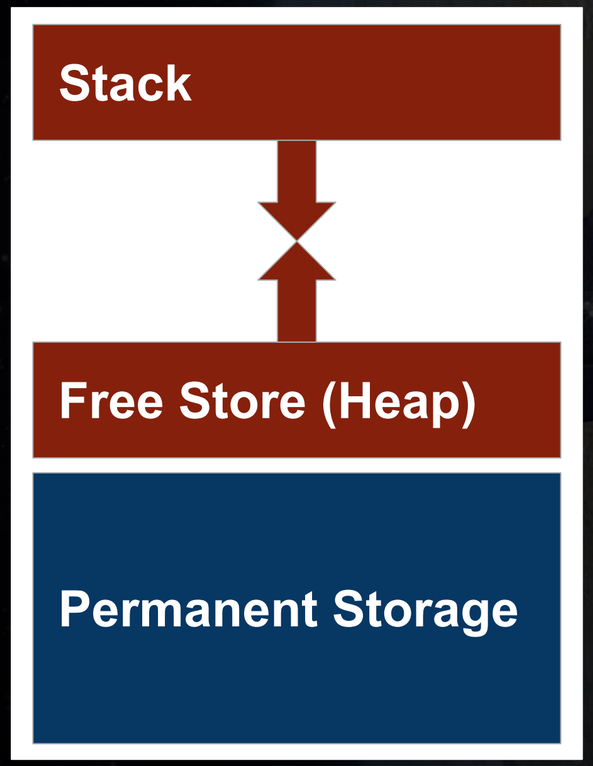
Accessors & Memory Layout



Accessors & Memory Layout

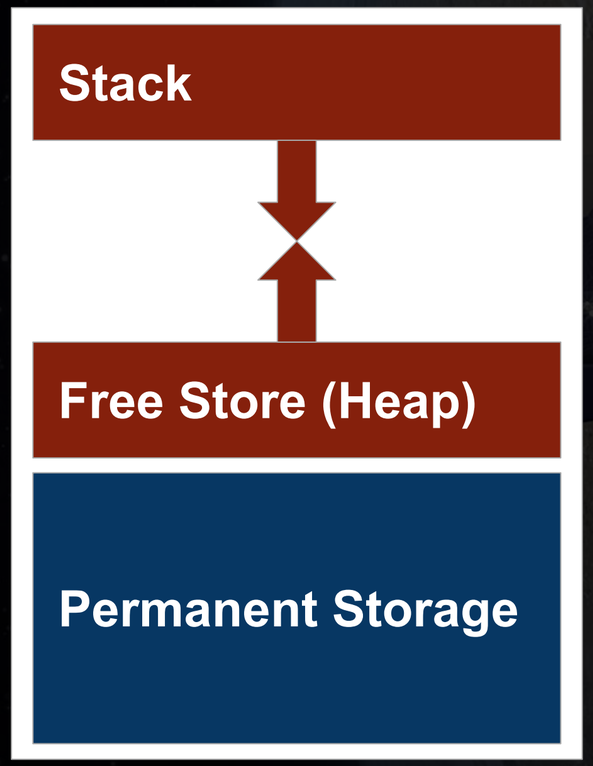


Accessors & Memory Layout



Left Term	Accessor	Examples
Class, Enum, Namespace	::	UGrabber::Grab EWordStatus::OK std::cout

Accessors & Memory Layout



Left Term	Accessor	Examples

Instance or Reference	`.`	`MyGrab.Grab()`
		`MyBullCowCount.Bulls`
		`MyGrabRef.Grab()`

Pointer	`->`	`MyGrabPtr->Grab()` `MyGrabPtr->Reach`

Class, Enum, Namespace	`::`	`UGrabber::Grab`
		`EWordStatus::OK`
		`std::cout`

Create a `Release()` Method

- Follow the example of the grab binding.
- The enum for release is `IE_Released`.
- Log that the key has been released.
- Test then jump with joy.

GameDev.tv

Reducing Code in “Hot Loops”

gdev.tv/uc2slides

Code That Runs Often

- A “hot loop” is code that get called often.
- TickComponent is a good example, every frame.
- Beware of code that you know will be called a lot.
- Make it clear what happens every tick.
- Refactor our code for speed...
- ...and make it ready for the physics handle.

GameDev.tv

Using Physics Handles

gdev.tv/uc2slides

Picking Things Up

- Unreal provides a Physics Handle that’s ideal here.
- Read the documentation.
- Get the physics handle working.

GameDev.tv

Refactoring Rules

gdev.tv/uc2slides

In This Video...

- Using multiple getters for multiple return values.
- Less lines of clear code is better....
- Naming is really important, take the time.
- Comment the “why”, don’t assume it’s obvious
- The “what” should be obvious...
- … but it can be helpful to add clarification.

Red, Green, Refactor

- **Red** - It’s not working (test failing)
- **Green** - It’s working (ugly is OK)
- **Refactor** - Make it pretty (must still work!)

Then you repeat the sequence.

Refactor Your Code

- Refactor your code.
- Yes it is soon, but clarity is worth fighting for!
- Make it so clear you’ll remember in a year.

GameDev.tv

Iteration Through Actors

gdev.tv/uc2slides

In This Video...

- I’ll show you how to access certain information.
- Then you’ll be challenged to iterate over a for loop.

Count The Total Mass

- Stop the player “rolling” around.
- Create an `OUT` macro to markup our code.
- Use a `for` loop to iterate through and add up the Actors masses.
- Make the mass to open the door a parameter we can edit in the editor.

GameDev.tv

Pointer Protection Process

gdev.tv/uc2slides

Protect All Your Pointers

- Horrible crashes when we follow a `nullptr`.
- We must always check pointers before use.
- When declaring always initialise to `nullptr`.
- Look for `*` in your `.h` files to help find pointers.
- Also check before every use and handle `nullptr`.
- Sometimes we may choose not to, e.g. Owner.

Go Forth And Protect.

- Check `OpenDoor.h` and `OpenDoor.cpp`.
- Log a helpful error if it’s null.
- Test that it works.
- Initialise any other uninitialised pointers.
- Make sure all pointer usages are protected.

GameDev.tv

SFX & Audio Clips

gdev.tv/uc2slides

In This Video...

- Setup an Audio component.
- `UAudioComponent` is the Component Type
- Calling Play on that when the door is opened and closed.

Challenge

- For consistency and readability put the `PressurePlate` log in its own function.
- Make the sound play once on `OpenDoor()` and once on `CloseDoor()`.
- Possible Hint: Use at least one binary switch to track if the sounds has been played.

GameDev.tv

?Optimisation And Student Questions?

gdev.tv/uc2slides

GameDev.tv

Building Escape Final Challenge

gdev.tv/uc2slides

Build A Level

- Create a level.
- Let’s make something more than a tech demo…
- Get some assets from the asset store.
- I’m going to use the Medieval Dungeon pack.
- Test your level design skills!
- Show off your creations in the discussions.

GameDev.tv

Building Escape Wrap Up

gdev.tv/uc2slides

In This Video...

- Congratulations on another complete section
- You’ve learnt so much, look at the lecture titles
- Please carry-on a little on your own and share
- Attached are useful resources
- Start the next section as soon as you’re finished.

Colours Slide

user defined type - #4EC9B0

comment - #57A64A

keyword/built in type - #569cd6

control keyword - #C586C0

variable - #9CDCFE

function - #DCDCAA

string/character - #d16969

escape character - #d7ba7d