UNREAL FEST

Bali 2025

Levelling Up: Advanced Perforce Techniques for Unreal Engine Projects and Teams

June 26, 2025

Agenda

- Best practices for working with branches and streams
- Best practices for folder structure
- Using P4 Shelves
- Code Reviews and project workflows
- P4 Admin cost saving tips

Resources and Links

https://jase.town/unreal-bali-2025/

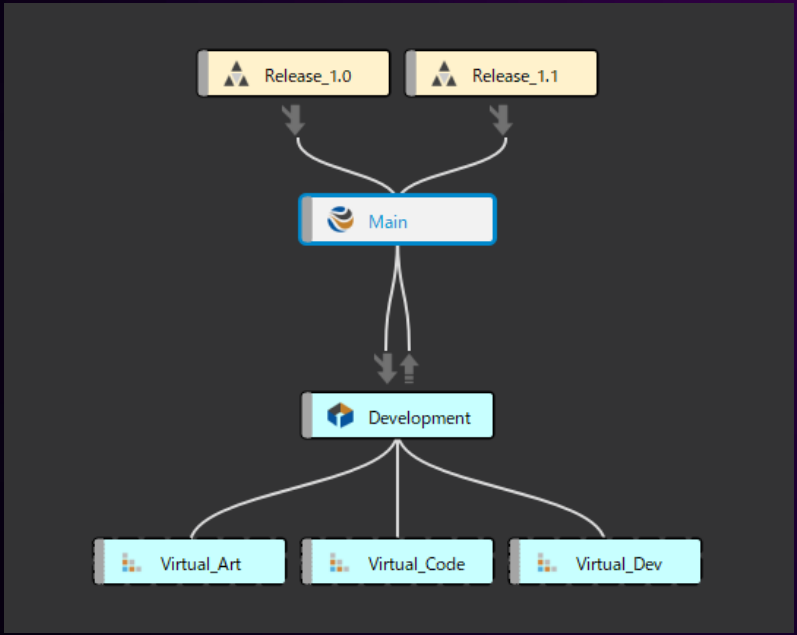
Why is it best to work in one stream?

Find Problems Early to *Fix* Problems Early

Rethink how you approach branching to keep your team on the same page.

Unified
Clear
Adaptable

What are P4 Streams and why are they useful?



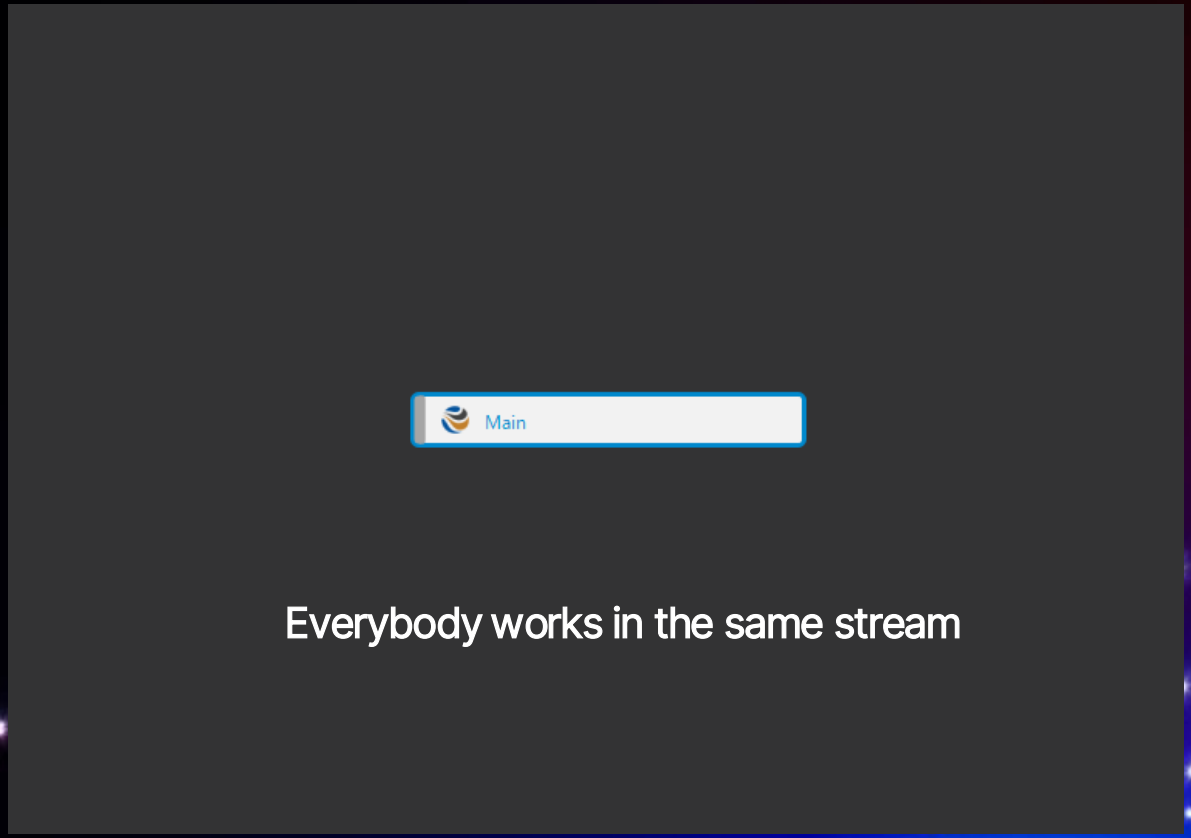
A Stream is Like a branch
(Except when its not)

A Stream defines a workspace view
Users can easily share views and switch views

Streams enforce merging rules
(But you can still break rules when needed)

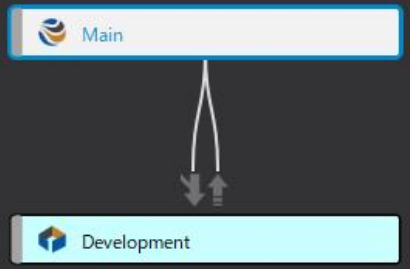
Streams have superpowers!
(imports, sparse streams, components, virtual streams)

Where your Stream setup should start



Only when you need to separate stable from WIP...

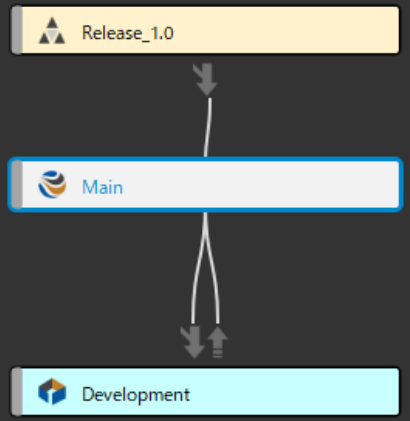
The project is copied up into Main
when it is stable for QA or Demos



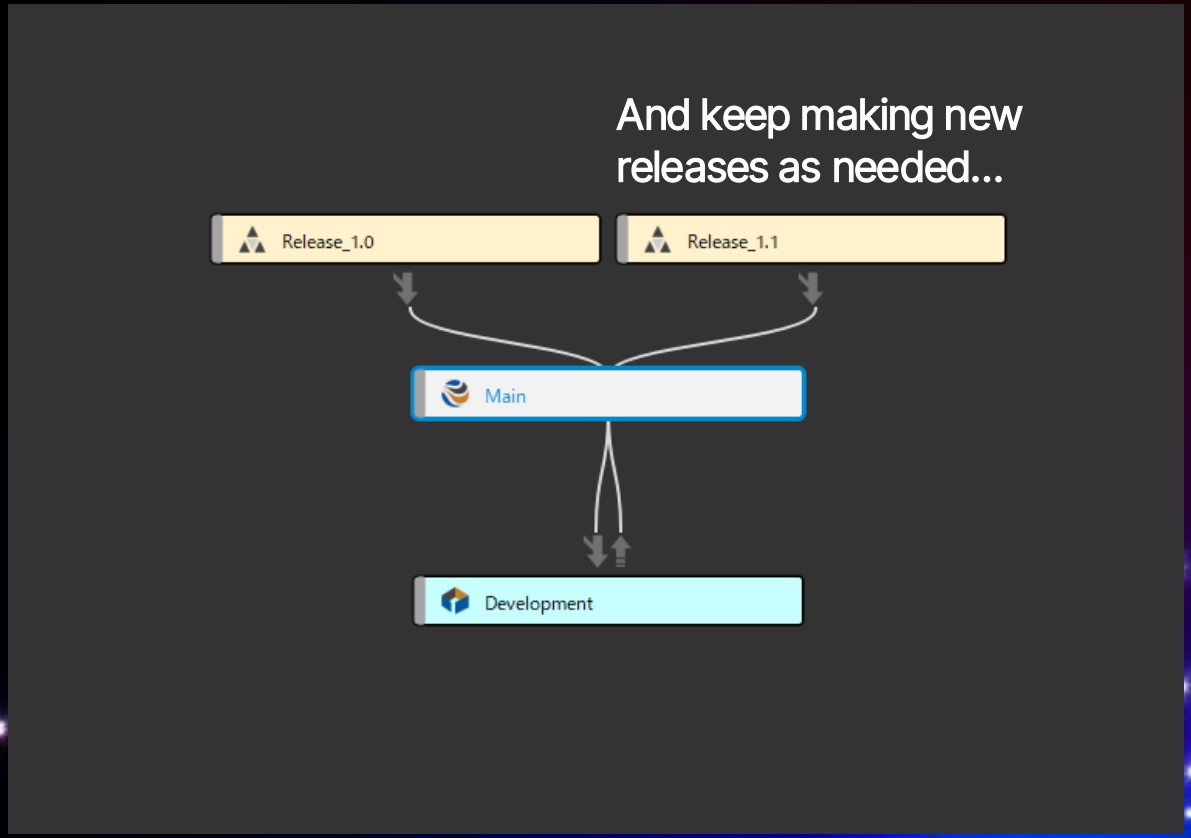
Work is done in the dev stream together

Releases allow you go freeze a release but still fix bugs

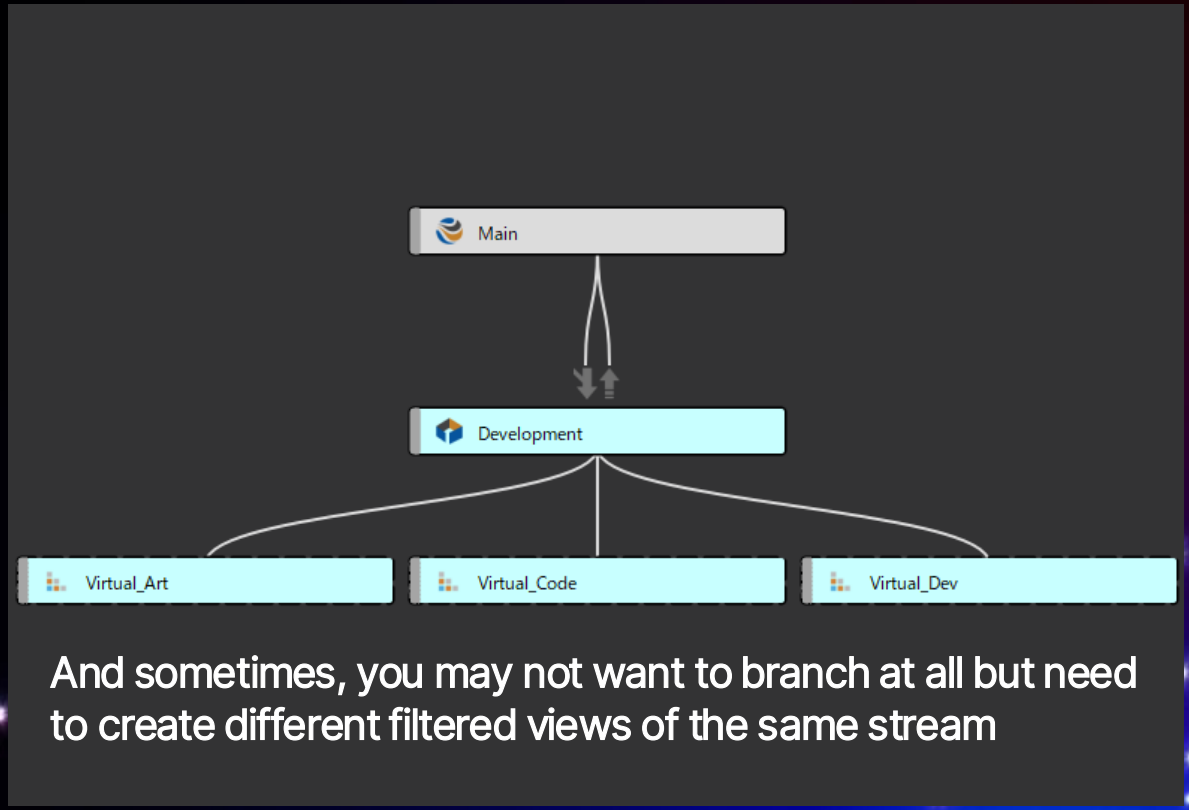
When it is approved, you can create
a Release stream to save that state!



Releases allow you go freeze a release but still fix bugs



Virtual Streams make it easy to give a limited view of one stream for artists, coders, or other users



And sometimes, you may not want to branch at all but need to create different filtered views of the same stream

other_workspace_bali, ssl:p4osaka:1666, other - Perforce P4V

File Edit Search View Actions Connection Tools Window Help

Refresh | Get Latest | Submit | Checkout | Add | Delete | Revert | Diff | Timelapse | Revgraph | Cancel

e:¥Perforce¥other_workspace_bali¥DCC¥

Depot | Workspace

other_workspace_bali (virtual_audio)

- e:¥Perforce¥other_workspace_bali
  - DCC
    - Reaper
  - Imports
    - Music
    - SFX
  - .p4ignore #1/1 <text>

Files | Pending | Stream Graph

Picker

Selection root is //BaliProject/Bali_Main

Copy/Merge Hints: Files Only | Label: Name | Width: S L

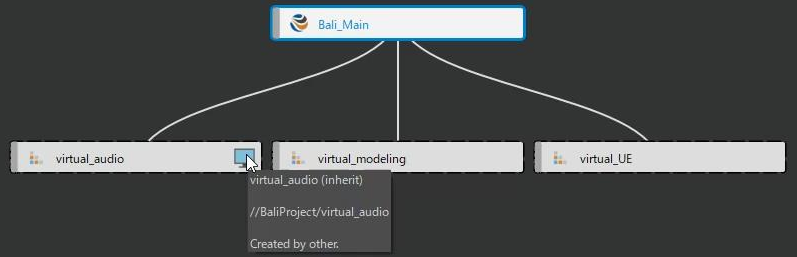
Depot: BaliProject

Mainline: Bali_Main (//BaliProject/Bali_Ma

Bali_Main

Apply

//BaliProject/Bali_Main



Log

Can't clobber writable file E:¥Perforce¥other_workspace_bali¥bali_project¥DerivedDataCache¥VT¥TEXTURE_7C16439390E24F1F9468894FB4D4BC55_VTCHUNK0393DE8C5B4673D99D1BECB92AA8C1F72FEB491

Can't clobber writable file E:¥Perforce¥other_workspace_bali¥bali_project¥DerivedDataCache¥VT¥TEXTURE_7C16439390E24F1F9468894FB4D4BC55_VTCHUNKD2DAB16BFF152507817E734DBB935160EA97A322

p4 sync -f e:¥Perforce¥other_workspace_bali¥bali_project¥DerivedDataCache¥VT¥TEXTURE_7C16439390E24F1F9468894FB4D4BC55_VTCHUNK02D4C3BCAD7DFA48AB32B8281FB06B8B83DB6275#0 {4 more items}

5 files removed

p4 sync --parallel=0 e:¥Perforce¥other_workspace_bali¥bali_project¥...#0

no files updated

1 warning reported

e:¥Perforce¥other_workspace_bali¥bali_project¥...#0 - file(s) not in client view.

p4 stream -o -v //BaliProject/virtual_audio

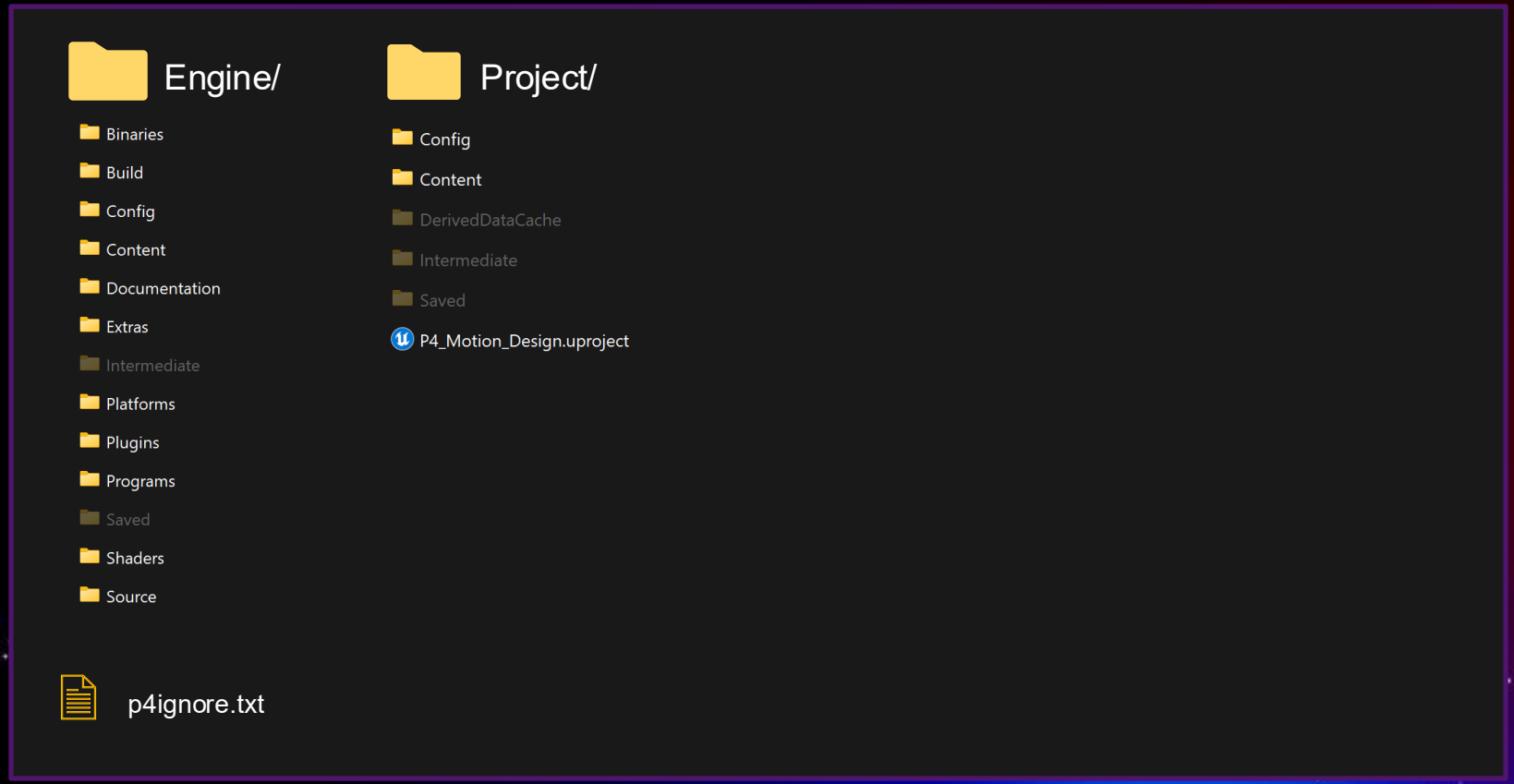
p4 streams -F Stream=//BaliProject/... | Parent=//BaliProject/...

p4 stream -o -v //BaliProject/virtual_audio

p4 stream -o -v //BaliProject/Bali_Main

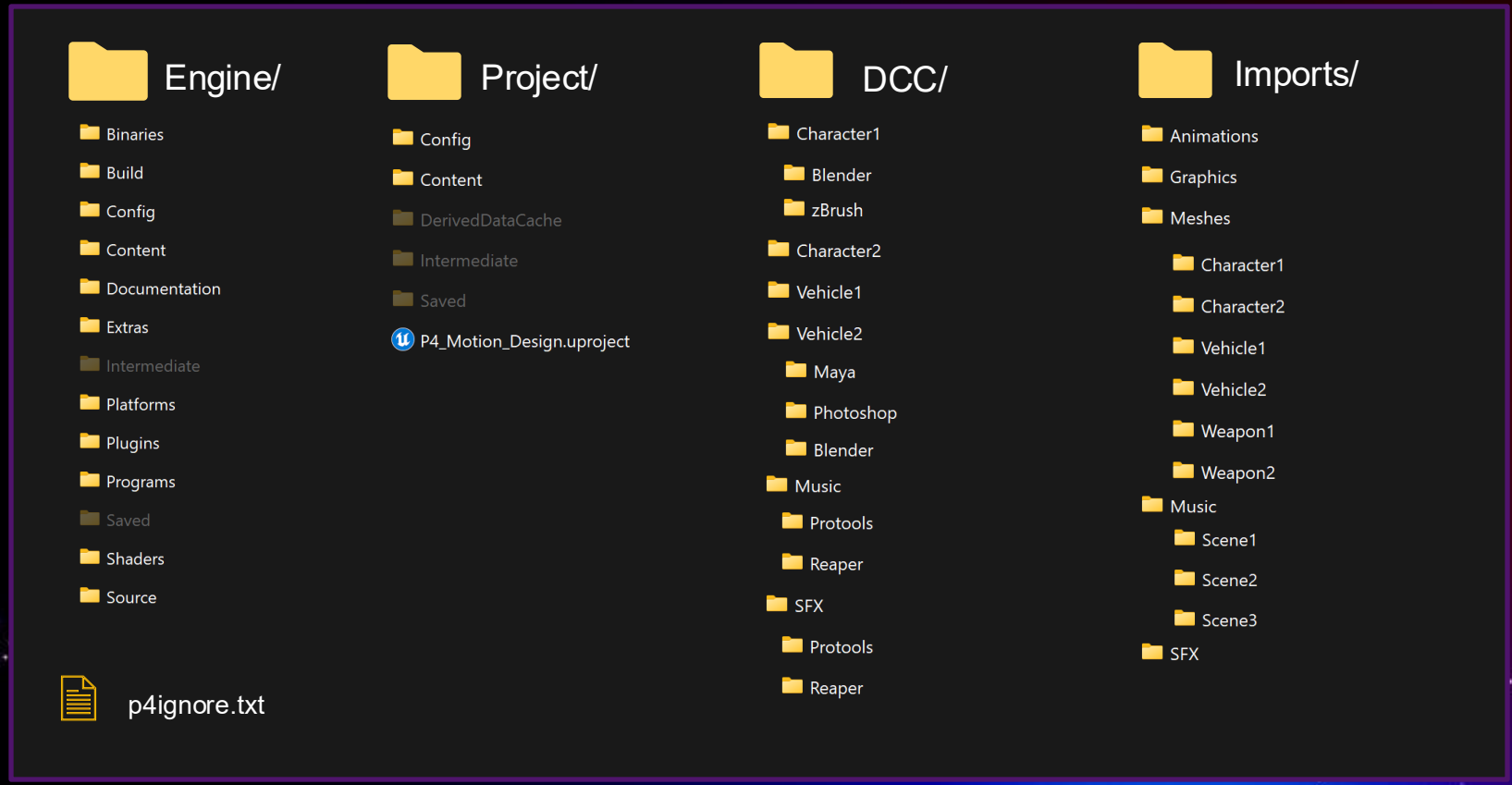
e:¥Perforce¥other_workspace_bali

Unreal Engine “Native” Folder Structure



Unreal Engine “Native” Folder Structure

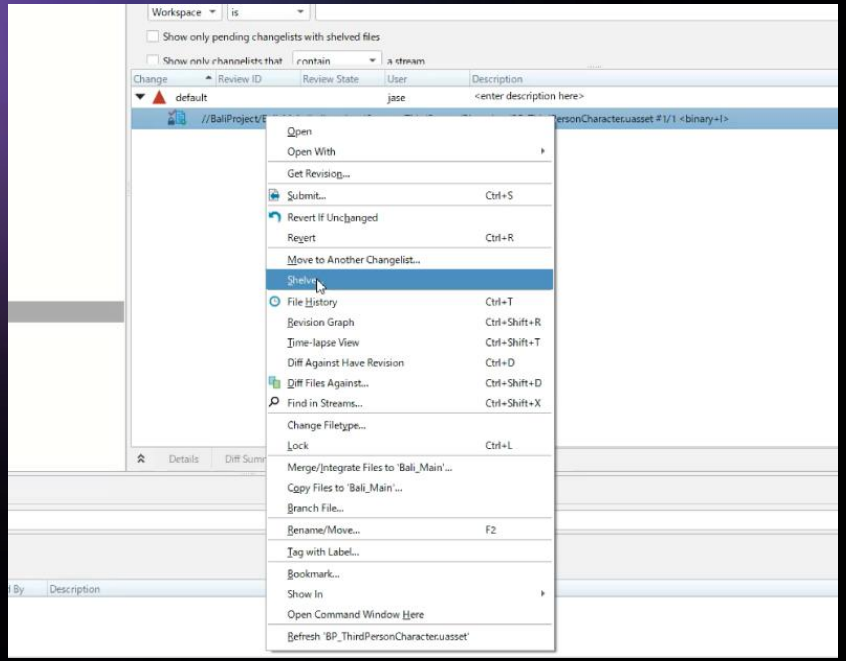
Plus Art Assets and Imports



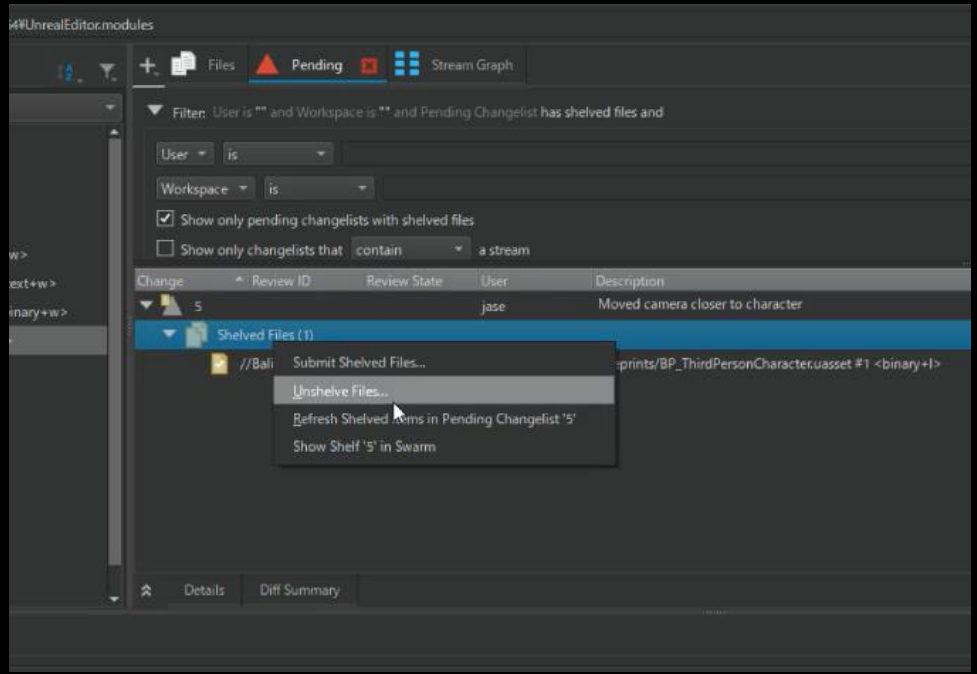
Leveling up workflows

Ensuring quality with Code Review

Step 1:
Using Shelves to share files before submitting.



One user can shelve…



And another user can access that shelf to review!

Step 2:
Using P4 Code Review to manage approvals

Pre-Commit Reviews

- Review shelves before they are submitted to the depot
- Ensures code base remains stable for the rest of the team
- Integrates well with automated tests and builds
- Generally produces higher quality code
- Can slow down work if the review process is slow or not enough reviewers
- Especially good for code, plugins, and essential functionality blueprints

Post-Commit Reviews

- Review changes after they have been submitted to the depot
- Faster paced
- Can create reviews any time in the future
- Removes the ability to run builds/tests before others can see changes
- Can be more difficult to fix complicated problems after they are submitted
- Can be a good fit for art teams

Create a Project in P4 Code Review

Create as many projects as you want
Projects and Depots are separate concepts

Set up “Branches”
These are any arbitrary depot paths you want to monitor in that project

Add Members, Owners, and Reviewers
Ensure all of your team is included in updates

Set Up Defaults for the Project
Defaults can include reviewers, workflows, test and deployment pipelines.

Set Rules with Workflows

Create Workflows and Use Across Projects

Once you define workflows, they can be used in any projects or branches that you want.

Workflows Enforce Rules

They determine if changes can be submitted

Automated Tests Can Be Added per Workflow

Automated builds and testing can save you LOTS of pain later on.

User Examples of Workflows

Code

- Requires review and approval before it can be submitted.
- Ensures quality code.
- Avoids pushing errors to the entire team.

Art

- Less likely to break everything so it can be submitted right away.
- Easier to review in context by Getting Latest
- Auto-generating reviews helps leads to see what was done during the day and to sign off on it.

Automating Tests and Builds

Set Up an HTTP Endpoint
P4 Code Review sends a POST request with data about the review that you choose.

Respond with Results
The Test automation responds to Code Review with a status and a message when done.

Can be Automatic or Manual
Automatic tests ensure higher quality, especially if they can run fast enough so you don't turn them off!

Tests Can Be Required to Approve
Requiring that tests pass before approving a review is a good practice so one submit doesn't cause others to fail.

Further possibilities

JIRA Integration
Link reviews to Jira tickets

TeamCity Integration
Automate your builds with TeamCity

Jenkins Integration
Automate builds and tests with Jenkins

The sky is the limit!
The P4 Code Review REST API allows you to customize as much as you want

Three tips for saving money on your cloud servers.

Save admin time with the Server Deployment Package

Faster deployment, easier upgrades, default configurations, standardized folder structure and naming

https://swarm.workshop.perforce.com/projects/perforce-software-sdp
(link available on resources page)

Consider Your Storage Volume Types

- SSD is the fastest and most expensive
  - Very good for random read/write operations, like databases
  - Use it for your metadata and logs storage
- HDD is usually about ½ the price of SSD
  - Slower for random read/write but decent linear read speeds
  - Use it for your larger depot storage
- S3 is much slower but much cheaper
  - Consider it for archive depots to keep your depot storage down

Watch out for Data Egress Costs!

- Cloud providers charge for all the data that leaves their cloud
  - Every time you sync large files (get latest) you add to this cost
  - Build machines that are on-premise can quickly add up if they always sync a fresh workspace
- P4 Proxy to the rescue!
  - P4 Proxy is super easy to set up (either with a package install or the SDP installer)
  - It caches synced files, so they will only be downloaded once
  - The proxy is invisible to the user, so you just connect to it instead of the main server and it works exactly the same (but faster!)
- Usually a proxy is all you need
  - And it is much easier to maintain than a replica or edge server
  - For large on-prem studios or users on multiple continents, however, edge servers can make a big difference

Thank you!