

Game Production Practice

Dr Robert Zubek, SomaSim LLC

CS 377: Game Development Studio
Winter Quarter 2024
Northwestern University

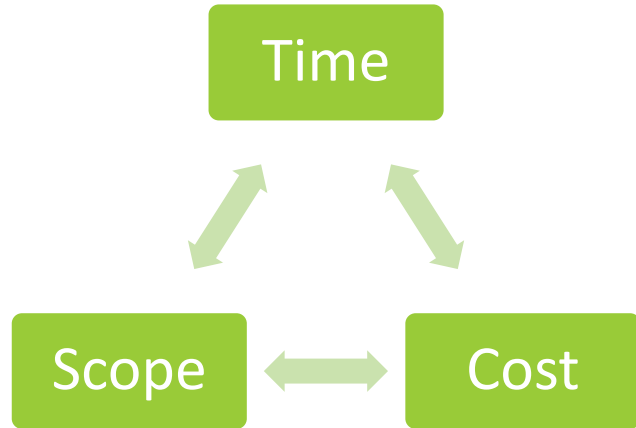


Today's schedule

1. Project planning
2. Weekly sprints
3. Playtesting and reports

End of class: Homework!

The project planning triangle



“What can you commit to?”
“Time, scope, or cost: pick two.”

Time

- When you have to be done

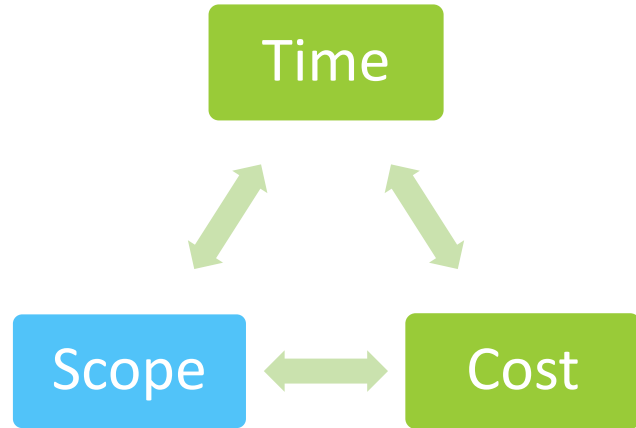
Cost

- Usually function of team size + external services

Scope

- Which features / how many / what quality level

The project planning triangle



“What can you commit to?”
“Time, scope, or cost: pick two.”

Time

- Shipping date usually fixed

Cost

- Team size capped based on budget

Scope

- This is the most flexible part!

(Sometimes people try to cheat: increase scope, and then introduce crunch to hide increased cost)

So how do we plan out scope?



Plan out NOTHING
ahead of time!

Plan out EVERYTHING
ahead of time!

Improvise?

No methodology:

- Minimal planning
- Incremental implementation

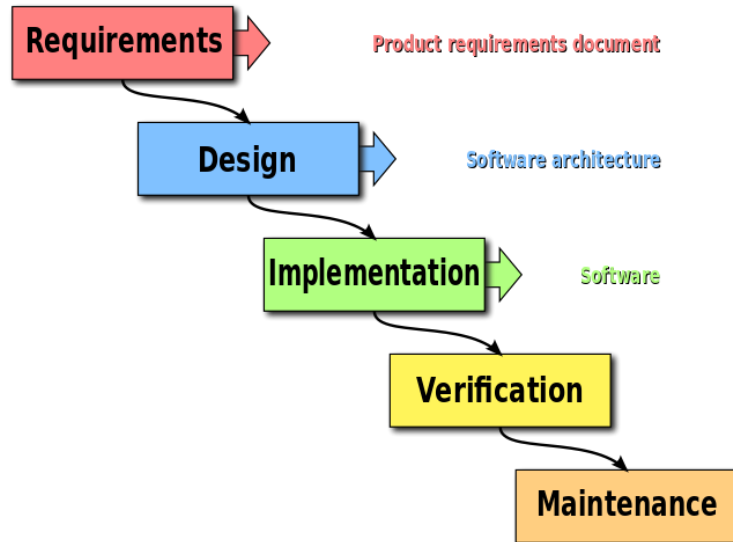
Problems

- Doesn't scale beyond a tiny team
- No plan => redoing things => wasted effort

... that said, that's how people often learn new things – by experimenting without a set plan



Plan out everything?



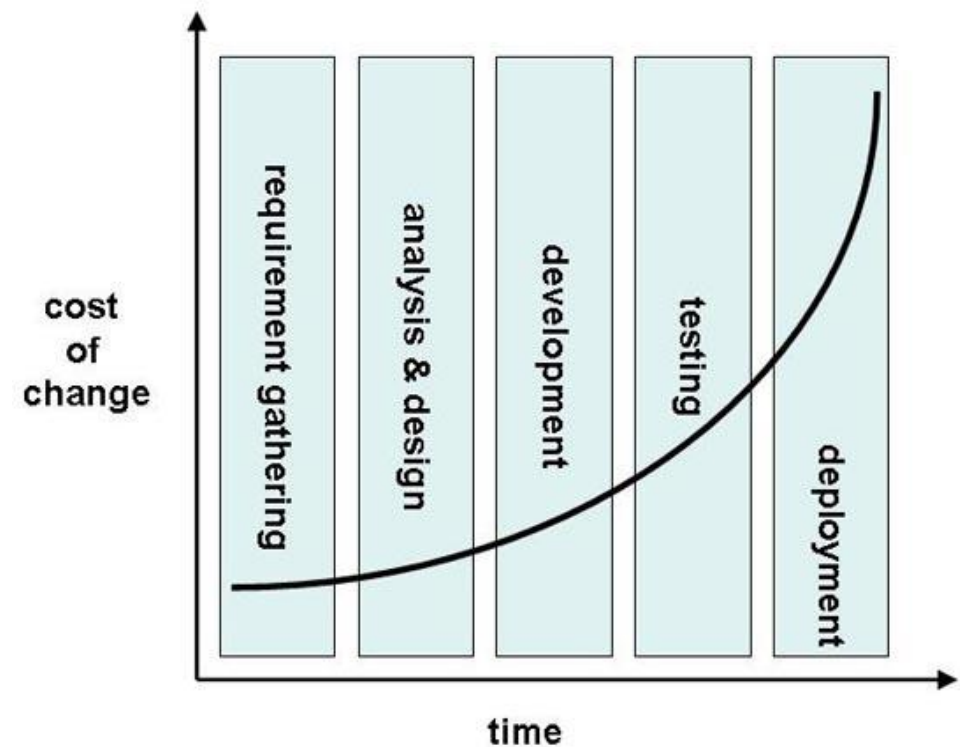
“Waterfall” model

- First figure out what the product needs to be
- Then design the software architecture on paper
- Then code everything up, test, ship, & profit!

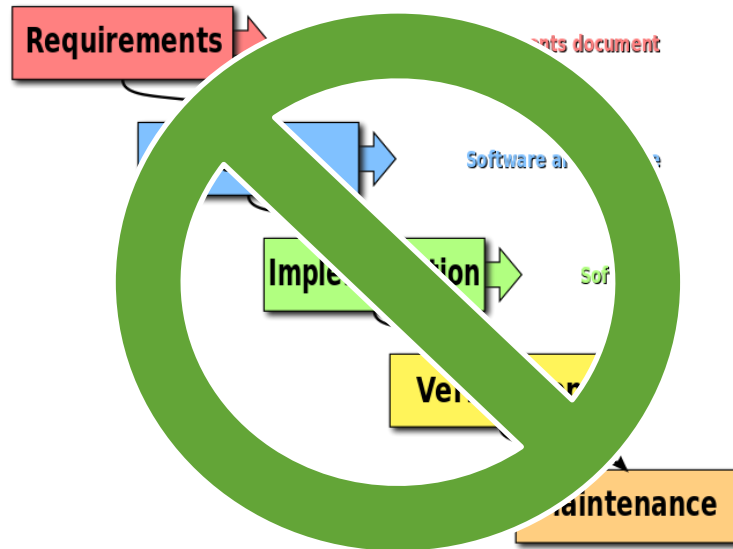
What makes waterfall appealing?

Cost of changes increases over time

Front-load all decisions!



Plan out everything?



Waterfall is a “strawman” model.
Actual designers don’t work like this.

- We don’t know what the goal is when we start
- We don’t know the decision space
 - It’s too large: we discover it as we make decisions
- We can’t evaluate individual decisions, only complete designs that bring them all together
 - Experience matters: prunes entire branches off of the tree
- Desiderata can change
- Constraints can change

Cautionary tale



(image credit: wikipedia @dmahalko)

Big company solution: greenlight process

Split production into stages

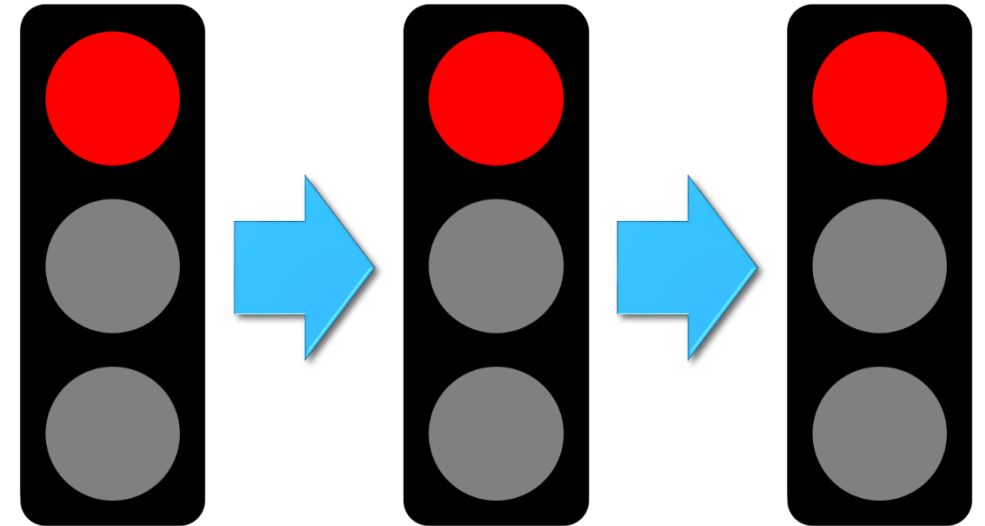
Start out with a blank slate and a tiny team

Several stages:

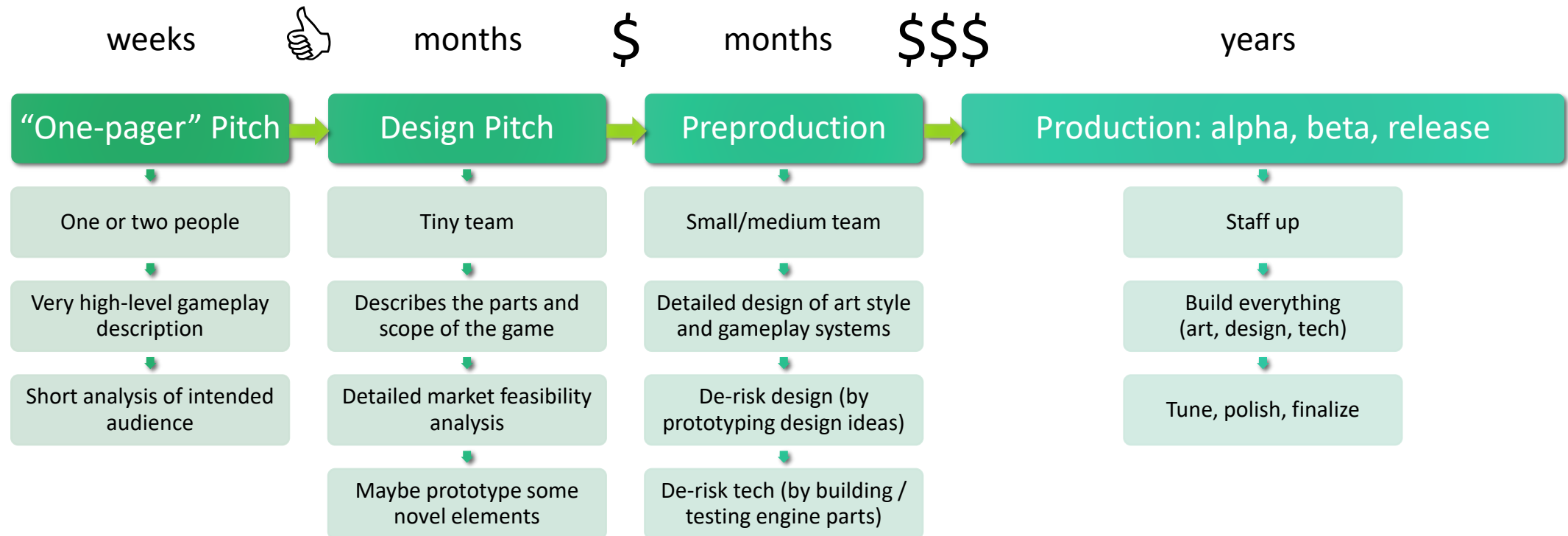
- Figure out more about the game
- Decrease decision space
- Increase resources

Final stage:

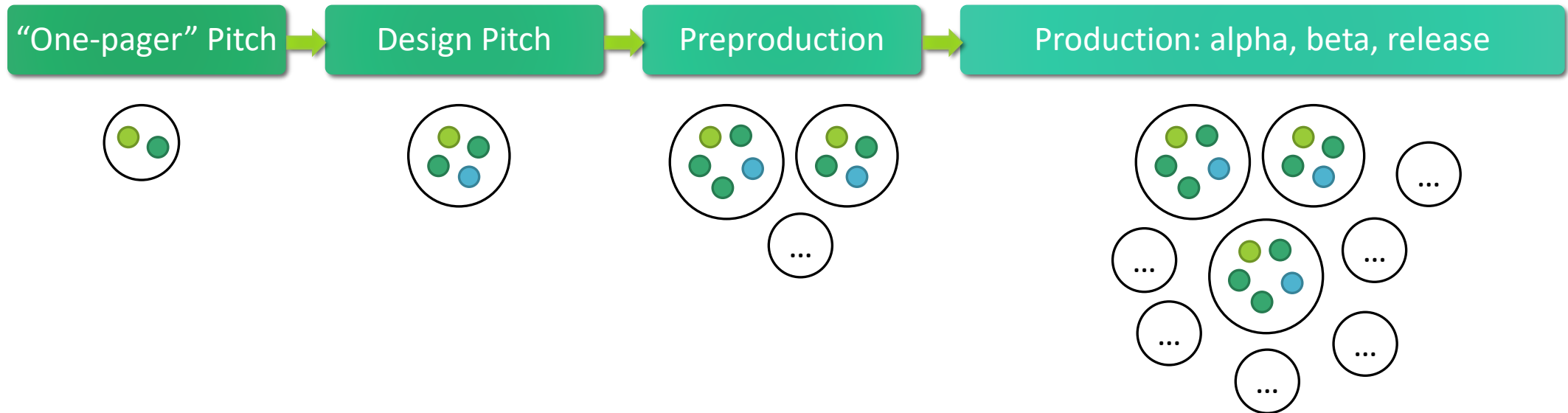
- Decide what the game is in detail
- Build the final version



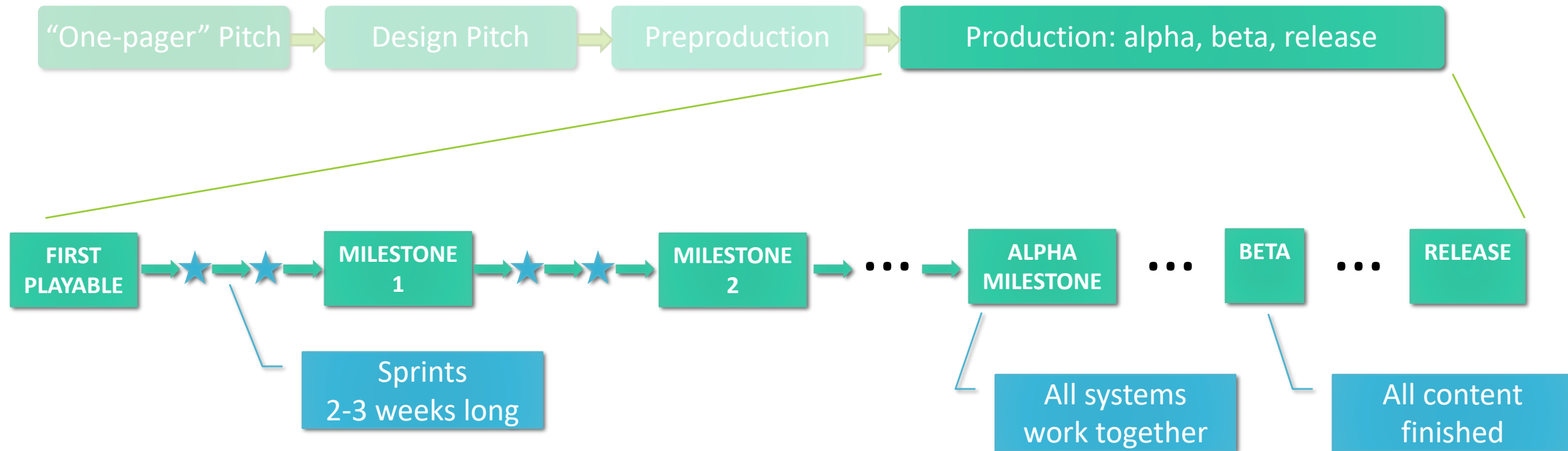
AAA production process



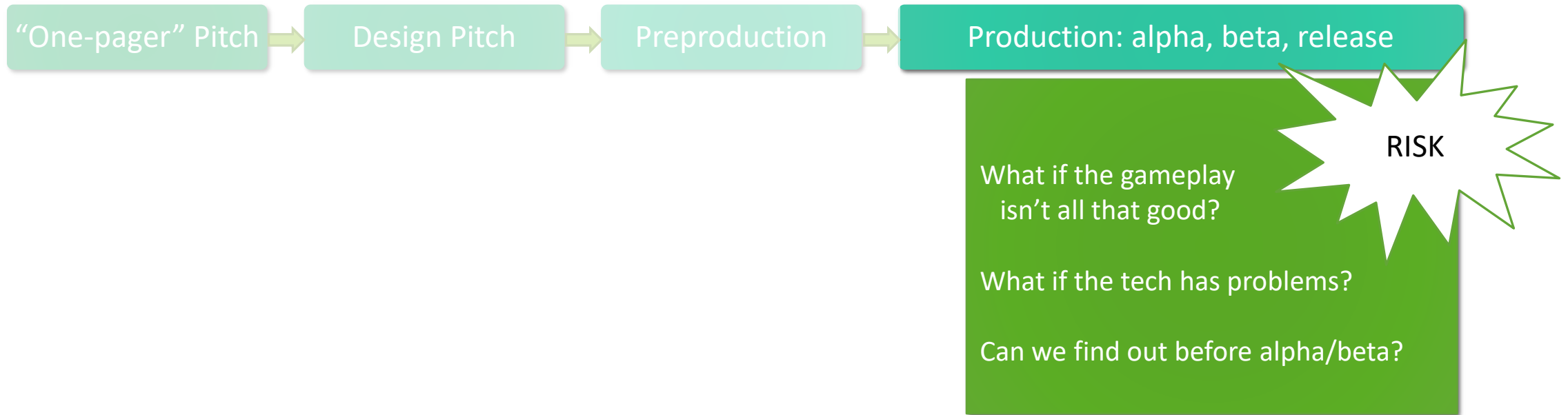
AAA production process



AAA production process



Quick aside on risk management



“Cerny method” for AAA



Goal: validate that all parts work together as a game, ASAP

“Vertical slice” – build a shippable-quality level of the game



Vertical slice

	Level 1	Level 2	Level 3	etc
Level design	+			
Art	+			
Story	+			
Game Design	+			
Tech and engine	+			



From AAA to smaller teams

AAA process for large games tends to be more rigid

- “Spiral” early on, but lock in as the team grows
- Optimizes **efficiency**, but limits **flexibility**

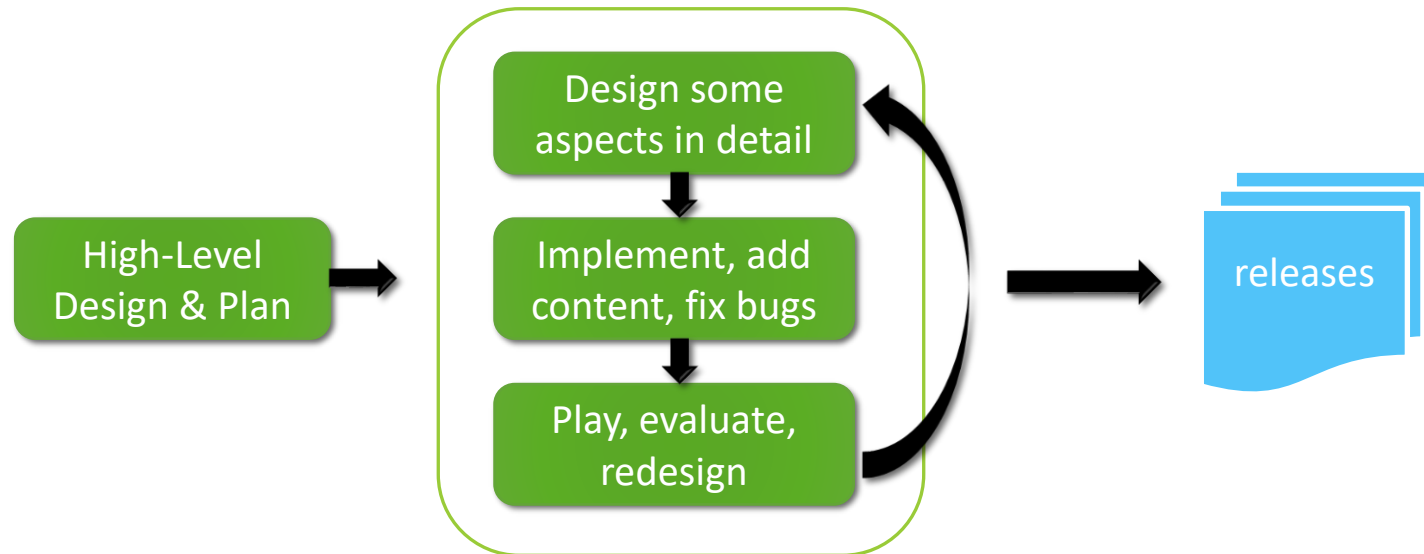
Small teams can be much more flexible

- Keep the iteration spiral going through development
- Adjust plans as you learn more about the game

Both large and small teams use some variant of “Agile” – we’ll be using it too! 😊

Small team / “indie” process

With a small team, you can get away with being more iterative
(Big teams would like that too, but it doesn't scale)



Gamedev version of Agile

Project → multiple milestones

- Rough division of what needs to be done when

Milestone → multiple sprints

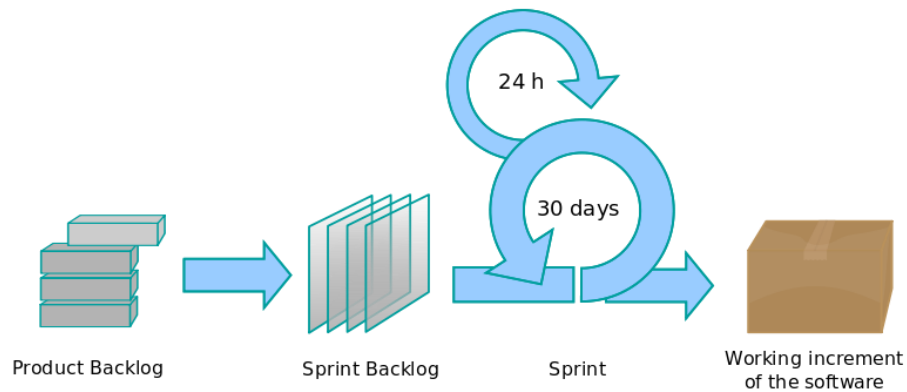
- Each sprint with a major theme

Sprint = predefined short period (e.g. 2 weeks)

- Sprint planning: team agrees what will be done
- Devs do their work, with daily check-ins
- Sprint ends on schedule. **No extensions!**



Sprint



Timeboxed

- 1-4 weeks

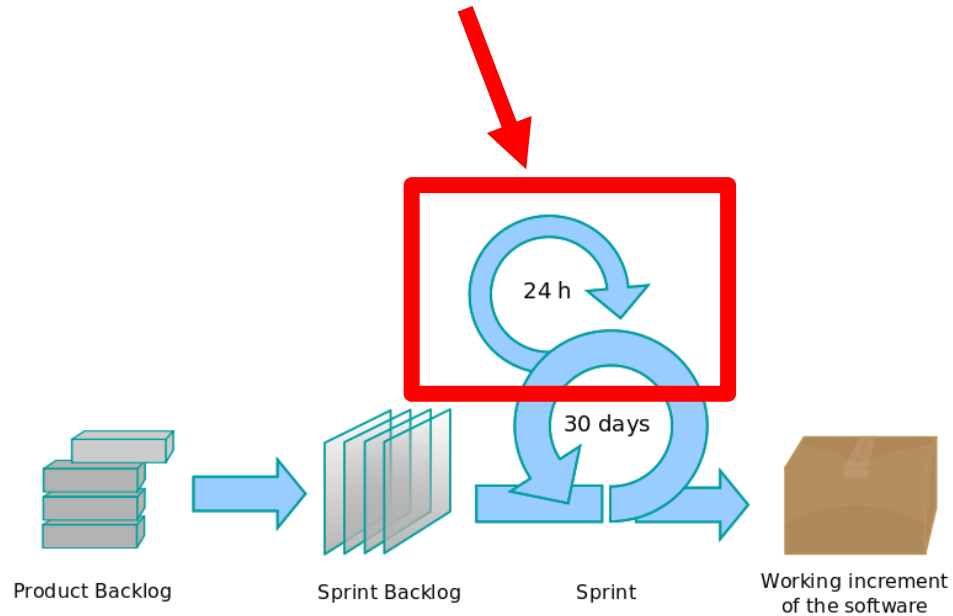
Starts with sprint planning

- Pick some goals, divide into tasks
- Devs do work estimates and discuss in group
 - Make sure this fits into the sprint!
 - Otherwise split it up into 2 work items, revise backlog

Ends with a new “product release”

- Features implemented to spec or not
- **But not half-done or buggy**
 - Don't merge it in unless it's done

Standups



Timeboxed daily meeting

- Short: ~15 minutes
- Starts on time whether you're there or not

Every member states

- What you did yesterday
- What you're working on today
- What problems or blockers you're running into

“Stand up” meeting

- Hold it in front of the task board
- Force people to stand, to keep it short :)














Task board

Tracks the status of features

- In backlog / to be done
- Being worked on
- Completed
- Blocked!

Shows at a glance

- What remains to be done
- Who is working on what
- How we're tracking towards goals

	TODO	IN PROGRESS	DONE	BLOCKED
Alice				
Bob				
Carol				
Dave				

Task board

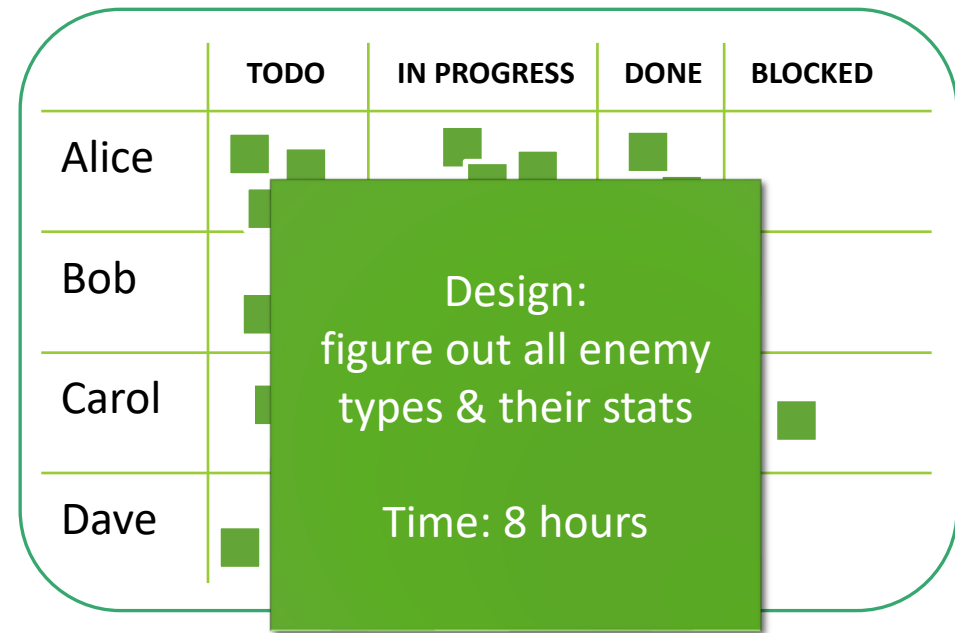
Ideally, each item has a time estimate

Used to see if it all fits into sprint schedule

- How many work hours assigned per dev
- If anyone is falling behind

Also done in software

- Trello, JIRA, etc.



Sprint deliverables

Commitment to Software Quality

Each sprint = software release

- Slowly growing the feature set
- No tech debt! “Shippable” code

Bugs

- **Fix as you go**, don't procrastinate
- Goal to have **zero P1 bugs at end of sprint**



Avoid thinking “I’ll fix this later”

Beware the “it’s **only a prototype**” rathole

- It’s common to cut corners
- And say “we’ll fix it for the final version”

A project’s **technical debt** is the sum of all those little fixes you have to do before you can ship

- All the workarounds and other extra effort you have to do because of it is like paying interest on the debt

“The **danger** occurs when the debt is **not repaid**. Every minute spent on not-quite-right code counts as **interest** on that debt. Entire engineering organizations can be brought to a stand-still under the **debt load** of an unconsolidated implementation”

Ward Cunningham (1992)

Example

You do a quick prototype of a new weapon in the game

And so you don't put in checks for invalid situations, so it throws exceptions and breaks the game sometimes

- It's just a prototype, right?
- You'll fix it later – **technical debt**

You still have to fix it sooner or later

- **Repay the debt**

In the meantime

- Playtesters will be distracted by crashes
- Your teammates' work is affected
- Lost work – **interest on the debt**

The longer you wait to repay the debt, the more interest your organization pays

Today's schedule

1. Project planning
2. Weekly sprints
3. Playtesting and reports

End of class: Homework!

Now let's talk about *your* projects...

How we'll run the project

1-week sprints

Standups

- **Twice a week** at least
- Before / after class, or however you arrange

Weekly sprint **reports** (one pagers)

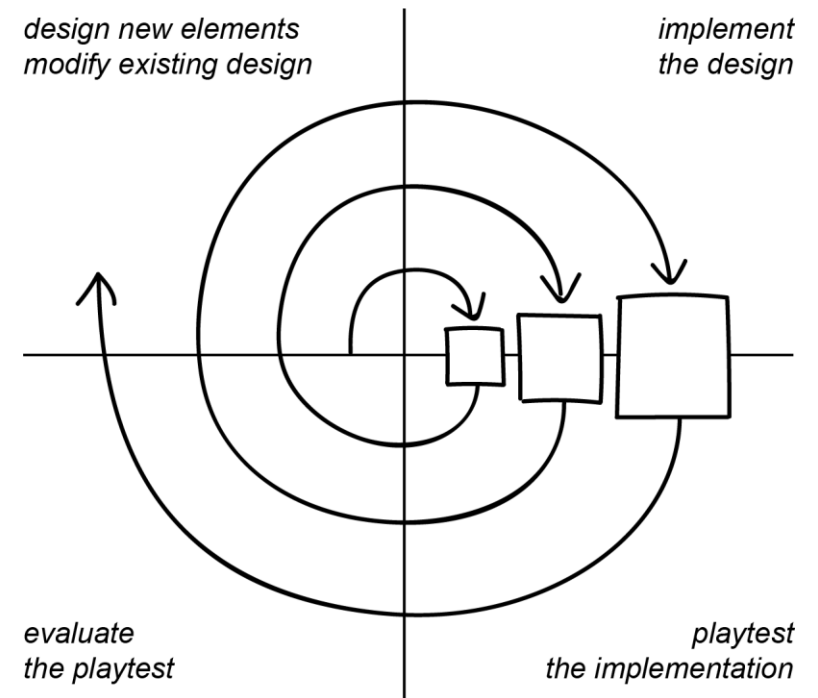
- Result of last sprint
- Goals for next sprint
- Backlog for next sprint with effort estimates
- Dates of standups + any notes
- → Due each week EOD Sunday

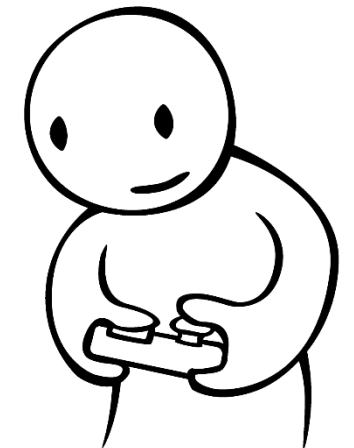
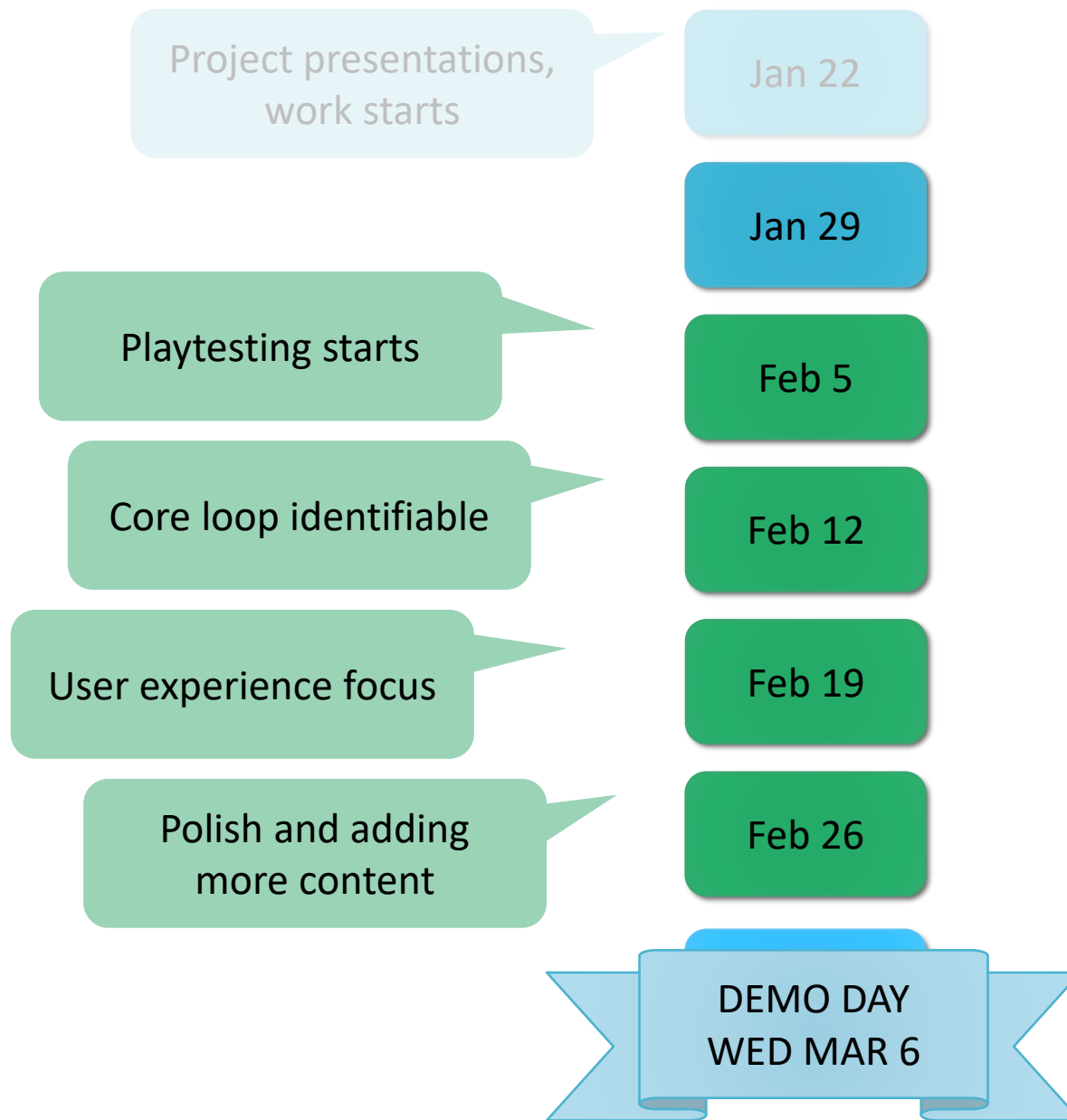
Weekly sprints and playtests

Each week you will:

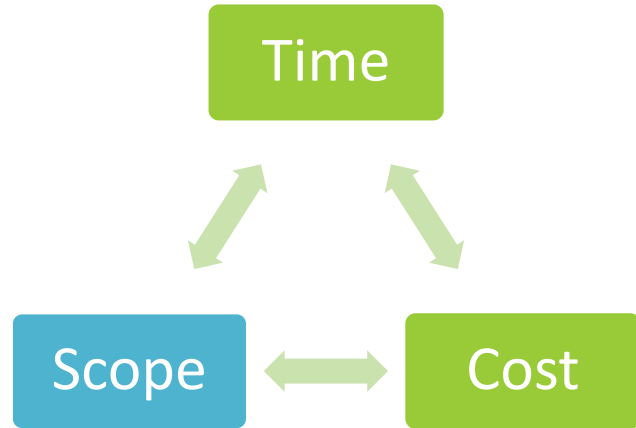
- Do a sprint with some specific goals
- Playtest whatever you're building in class
 - Get feedback from others
 - Give feedback on their work
- Integrate feedback into your planning for next sprint

→ Useful tactic: **“find the fun”**
(what are your playtesters really enjoying?)





Your class project



Time

- Deadline: end of quarter

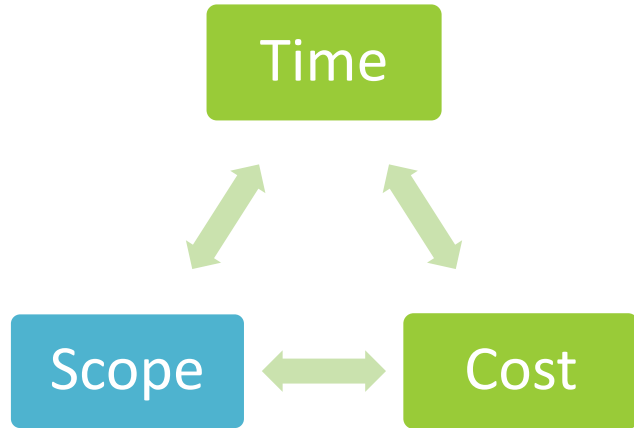
Cost

- Your team size is set

Scope

- That's up to you!

Your class project



Time

- Deadline: end of quarter

Cost

- Your team size is set

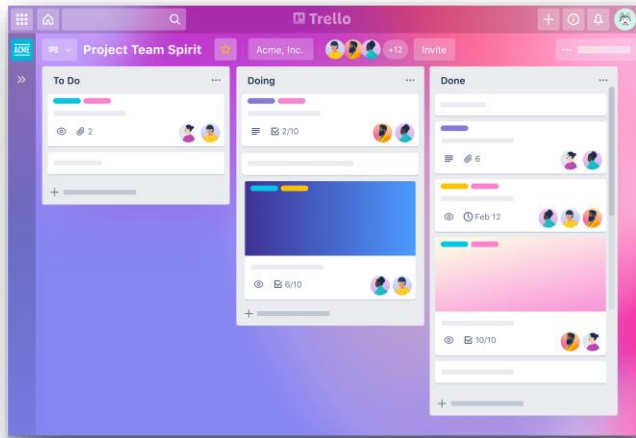
Scope

- That's up to you!

Make a list of the various game elements.
Consider their importance and cost.
Have a backup plan for scope reduction.

Remember - the goal is to
have a fun game by the end
of the quarter! Focus on core
loop & fun gameplay.

How we'll run the project



Tools

Task board

- Use Trello
- Maybe: Google Spreadsheets
- Or even a whiteboard 😊

Chat


- Discord? Probably the easiest

TrelloWorkspacesRecentStarredTemplatesCreate

BoardCity of Gangsters QAProject Highrise Console QAWorkspace visible

AutomationSlackFilterShow menu

Reported by QA



This is an example card

1

Master Ticket - Tutorial Issues

Master Ticket - General Issues

Master Ticket - Chicago map

Master Ticket - Cincinnati map

Master Ticket - Pittsburgh Map

Master Ticket - Detroit Map

Master Ticket - Bridie (Pittsburgh)

Master Ticket - Christian (Pittsburgh)

Master Ticket - Lynsey (Detroit)

Master Ticket - Aaron (Detroit)


Violet's Auto Body Shop

+ Add a card

In development

+ Add a card

Dev done, waiting for build



Clara Richardson
Local troublemaker: Richardson's Crew

Small-scale troublemaker who controls a single corner.

Interested in: BusinessStance:

Clara Richardson
You're reliable. You helped out my friend Jan Roth with a bit of labor the other day.

Herbert Allen
Got any leads on interesting goods or product? I'd like to do business with some of the people on your corner. I understand they're afraid of choosing you.

You know what? I don't like the way you're looking at me.

Forget it, I'll see you around.

I get a message about someone offering a reward but when I get to the corner goon, they never say anything about a reward, just the usual business.

Recent Transactions

Controlled Buildings

Known Fronts

Scheduled Deliveries

Car Shops

Police Precincts

Heat

Respect

Zoning

Population

American

Austrian

English

German

Hungarian

Irish

Italian

Polish

Quonian


Cincinnati - American nationality doubles up in population info overlay

Kept being told that one of my vehicles was getting banged up. Couldn't tell which one as nothing was showing up but when I took them all to the garage one by one, they were all fine.


Dead body tooltip changes depending on the previous tooltip consulted

+ Add a card

Ready for verification / retesting



The character's name overlaps with many other UI elements




Just a '?' in the conversation box

Distillation Moonshine - Level up error/typo

Avatar portrait appearance changes


TYPO - Extortion proposition dialogue



+ Add a card


Retesting passed

Block in the Loot screen with two cars on the same tile




"Start mission: ?" as a "reward" on completing a request

Closing the 'Select someone to talk to the business' window makes it so that window won't open next time the business is clicked on



Duplicated 'American' nationality on character creation when selecting Cincinnati as the map




No character limit when entering text on 'Random value', 'First name' and 'Last name' during character creation


+ Add a card

As intended - no work needed


White floor texture on some corners on loading a save




John Smith's avatar doesn't match his age



Butcher shop produces corn and steel barrels




Spelling mistake on Lesson 10: Operations (11/14)




0.30 units of moonshine on my

+ Add a card

For future consideration - no work needed



Orange highlight for on-map elements can be rendered over other interfaces



FBI Agent hasn't moved since start of game

Upgrade available takes my distillery to a lower tier

Framerate drops momentarily when checking 'All shared connections'

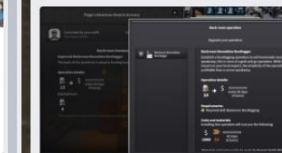
Multiple favours do not grant multiple relationship boosts with the same NPC

The settings can't be accessed or changed ingame

+ Add a card

Obsolete or development stopped

Multiple problems loading Detroit saves from a previous build (from 20210708 to 20210712)



Backroom Moonshine Bootlegger shows up as an upgrade after I have already upgraded to Improved Backroom Moonshine Bootlegger,

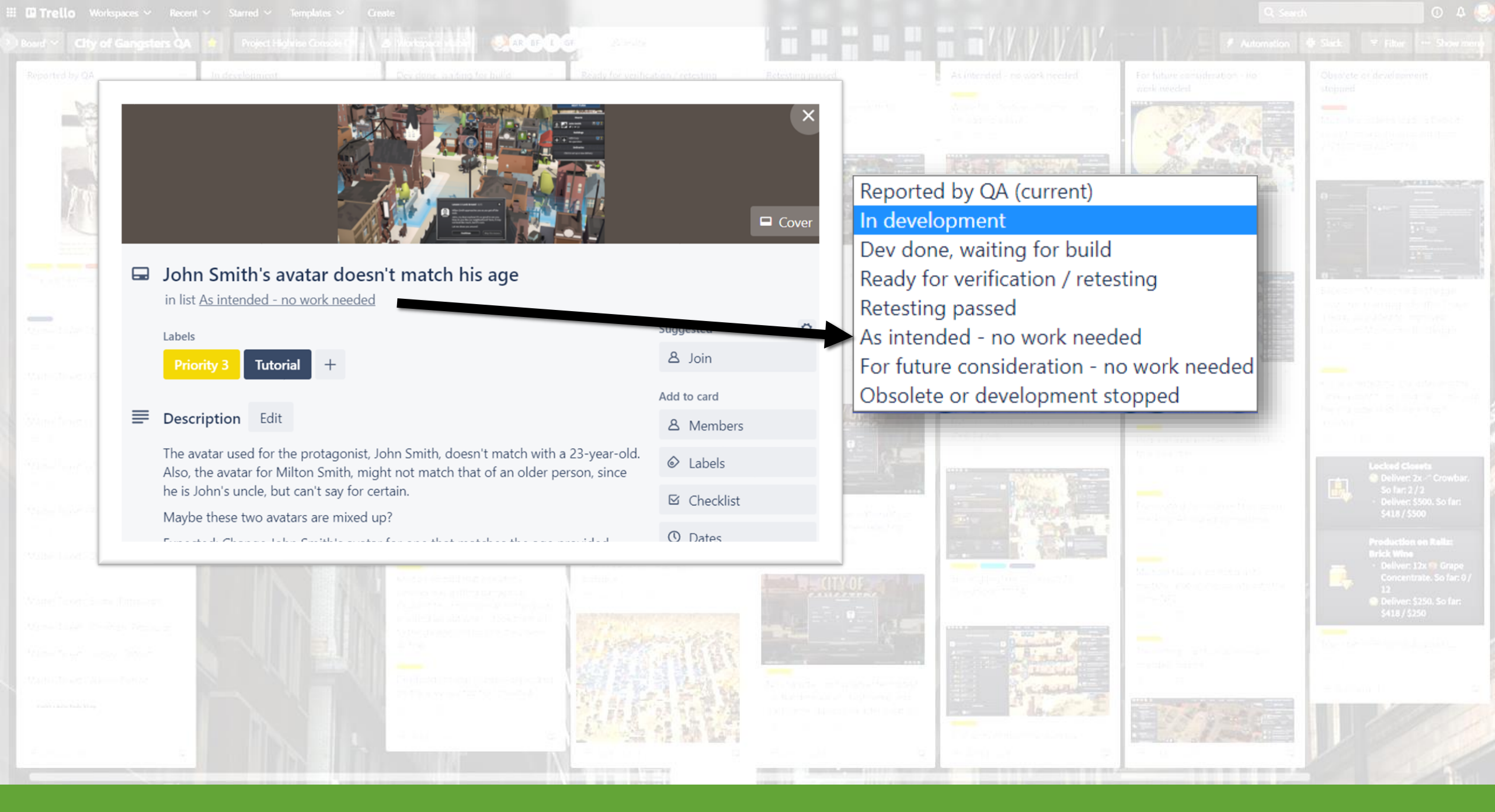
Police arrested my character and the camera went to my original corner, but the character is still were it got arrested

Locked Closets

Production on Rails: Brick Wine

Train station missions doubled up

+ Add a card



- Reported by QA (current)
- In development
- Dev done, waiting for build
- Ready for verification / retesting
- Retesting passed
- As intended - no work needed
- For future consideration - no work needed
- Obsolete or development stopped



TODO

IN DEV

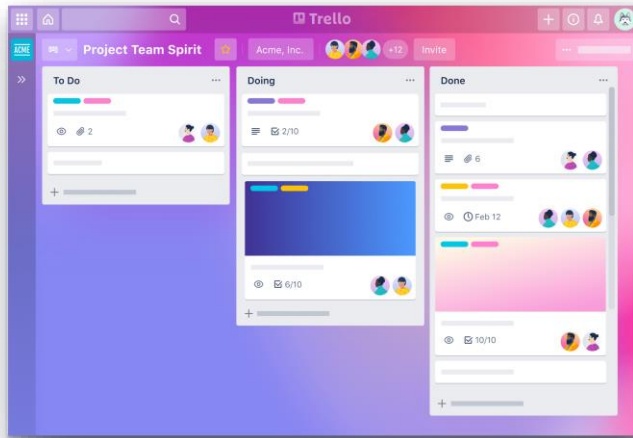
DONE

TO CONSIDER

WON'T DO

Start by making five lists

How we'll run the project



Tools

Task board

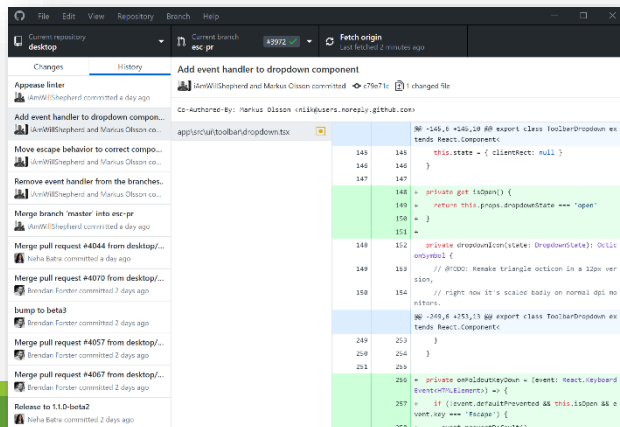
- Use Trello (maybe: Google Spreadsheets)

Chat

- Discord? Probably the easiest

Source control

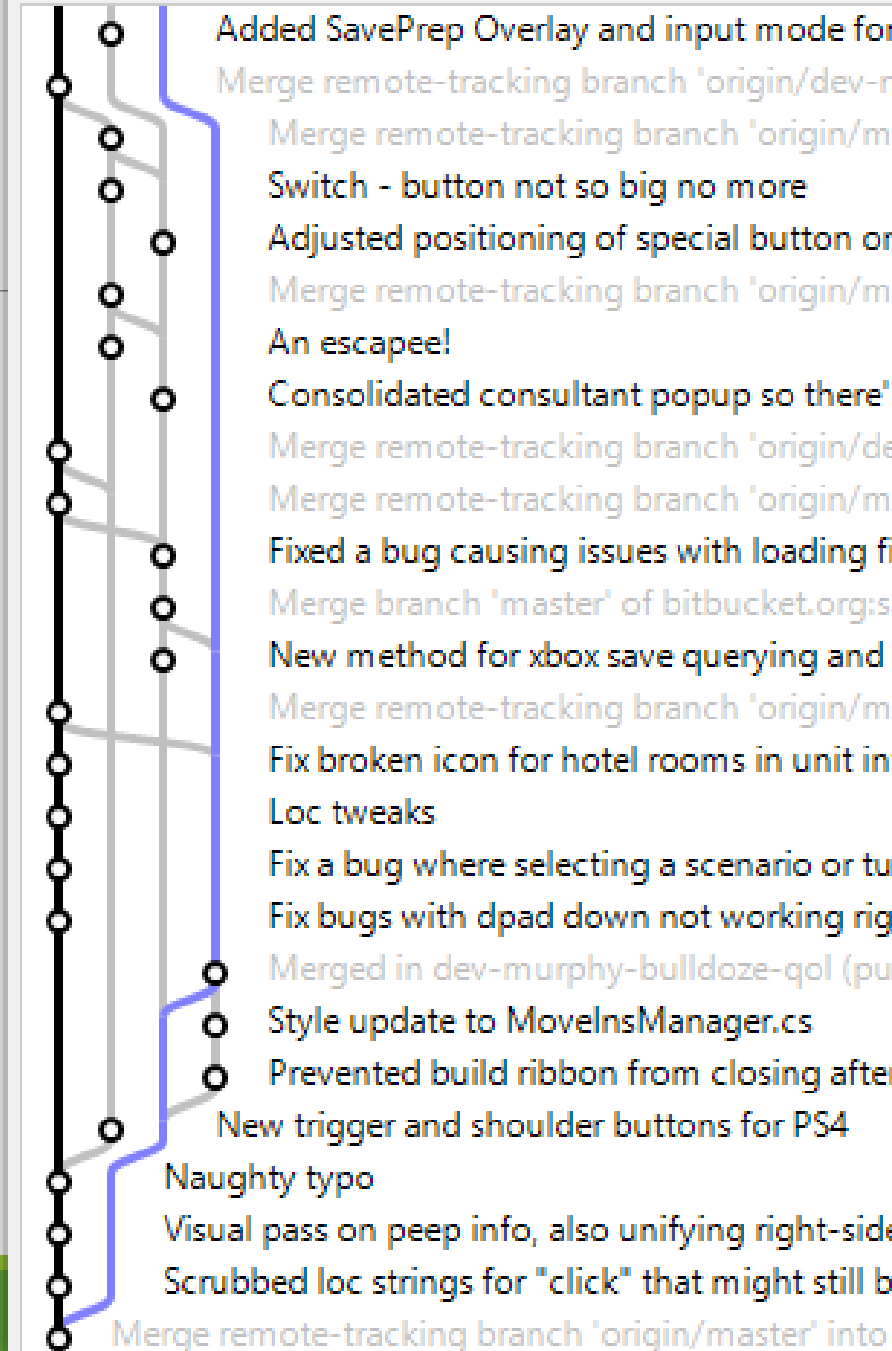
- Up to you (git, etc.)
- Hosting: GitHub, BitBucket, etc.
- GUI: GitHub Desktop is free



Source control tips

1. Use git with a GUI!
 1. <https://git-scm.com/downloads/guis>
 2. For small projects I like GitHub Desktop (free)
2. Use separate branches as needed

Commits

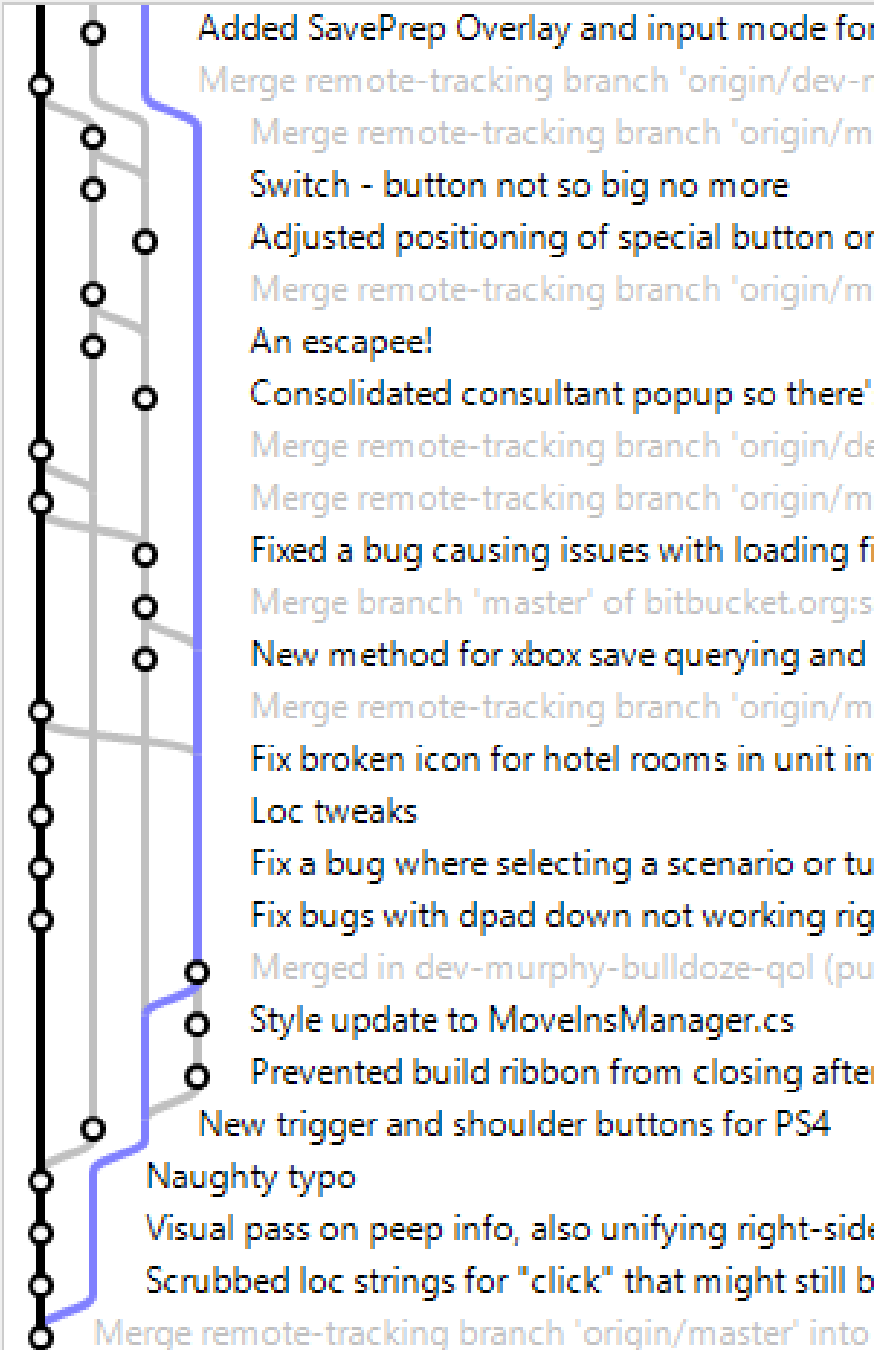


Source control tips

1. Use git with a GUI! <https://git-scm.com/downloads/guis>
2. Use separate branches as needed
3. Watch out for stuff that doesn't merge well!
 1. Source files, text files are okay
 2. "Assets" don't merge!
 - Scene files
 - UI prefabs
 - Art assets
 - etc.

COMMUNICATE OVER CHAT ABOUT
WHAT YOU'RE WORKING ON

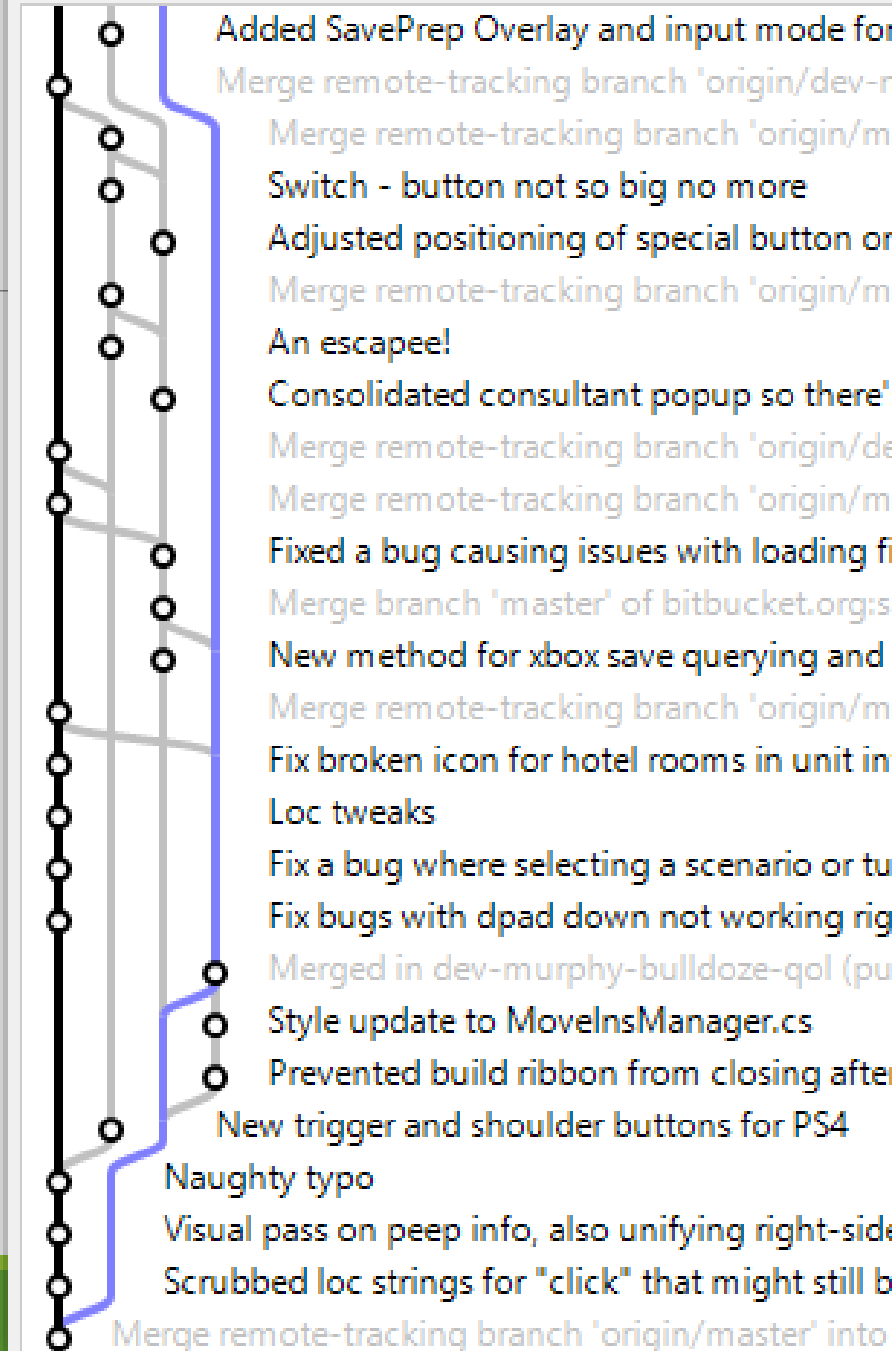
Commits



Workflow tips

1. Treat **main** as “always shippable”
2. Merge early, merge often (e.g. every day or two)
 1. Main should be “always shippable”
3. DON'T MERGE RIGHT BEFORE DEADLINE
 1. Seriously.

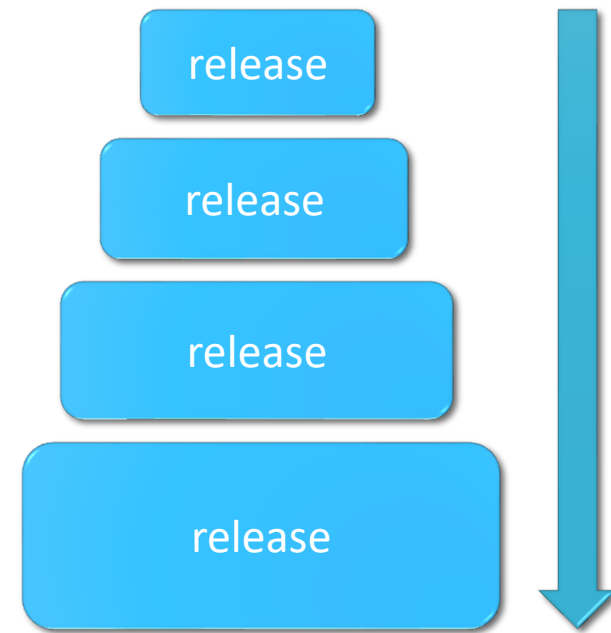
Commits



Sprint deliverables

Each week = software release

- Slowly growing the feature set
- No tech debt! “Shippable” code



Today's schedule

1. Project planning
2. Weekly sprints
3. Playtesting and reports

End of class: Homework!

How it works

Each week **all teams bring their games** to class, twice a week

Each week **~1/4 of you** will be assigned to **playtest** games that week

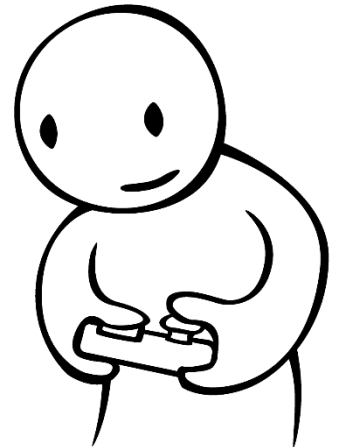
- I'll make a proper schedule soon :)

If you're on the dev team:

- Get feedback from testers to see how it plays and iterate

If you're playtesting a game:

- Give feedback while playing!
- Also send in a report.
- *More on this in a second.*

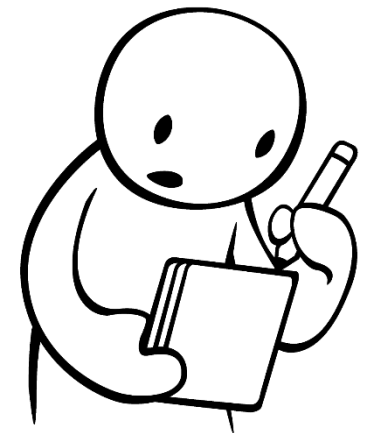


Demo Teams and Playtesters

Demo Teams will be **bringing to class**:

- A laptop running their game build
- Notepad or note-taking app

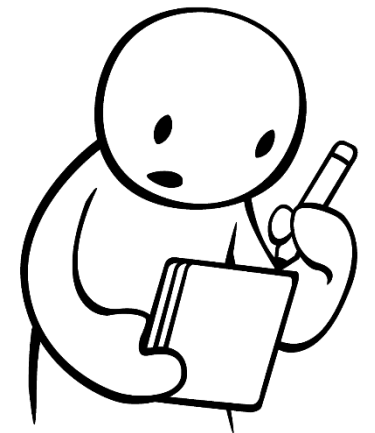
Playtesters will spend 10-15 minutes with each assigned team



Info for Teams

When showing your game:

- Watch the playtester and pay attention: **what do they like, dislike, what confuses them, etc.**
- After the playtest, you should have a **small list of ideas** or feedback
- **Triage the list:** split it into three groups:
1. yes we should do this, 2. let's think about it, 3. ignore
- This is for your **internal use**, to guide your work for the rest of the week



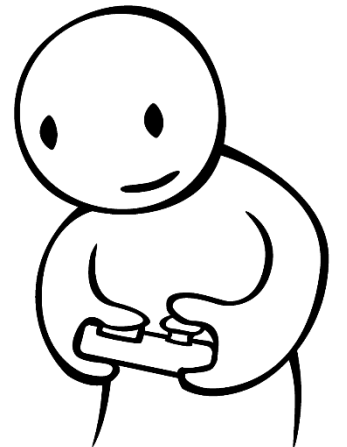
Info for Playtesters

Every week, $\sim 1/4^{\text{th}}$ of students will be picked to playtest games shown that week

- **Schedule will be posted later**
- **Give your feedback** to the team as you play!
 - Say out loud what you're thinking as you're playing

Ratings

- Give each game a rating on the scale [0, 200]
 - where 100 = good progress comparable to your own team
- Submit your ratings by EOD Sunday of your week (on Canvas)



Weekly reports

Weekly reports

Due **EOD Sunday**, every weekend between now and demo day

There will be three separate “assignments” on Canvas:

1. Team status update
2. Individual teammate assessment
3. *For those who playtested that week*, playtest scores

Team status update

One pager that states:

- Result of last sprint
- Dates of standups + any notes
- Triaged list of tasks from playtests
- Task items for next sprint with effort estimates (hours per person)

Teammate assessment

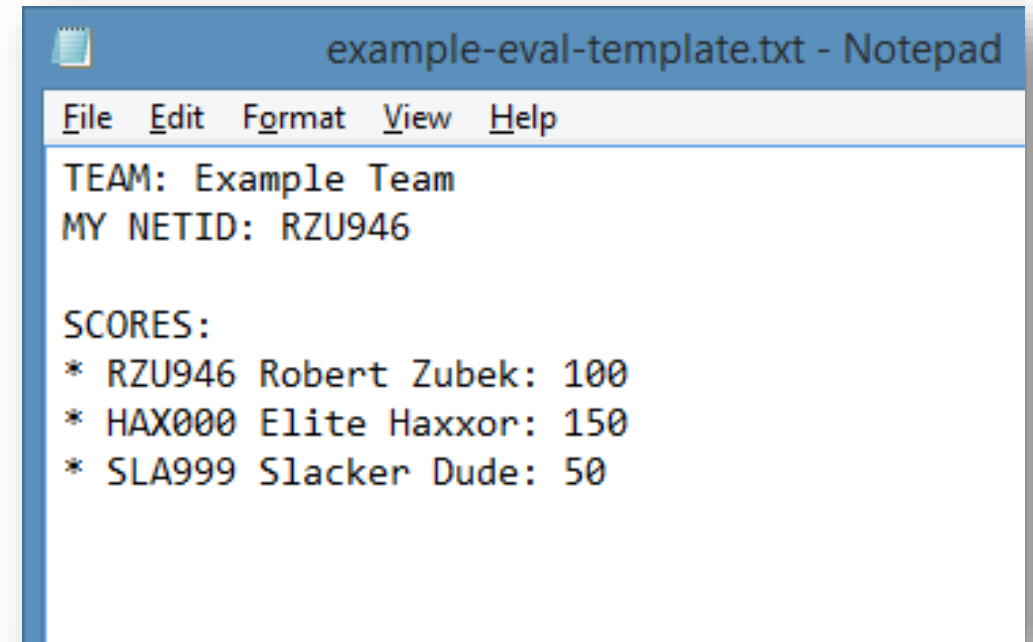
Weekly feedback on how other people in your team are doing

Rate your teammates on a subjective scale:

- 100 = contributing as much as I am
- 200 = contributing twice as much!
- 50 = contributing about a half

Submit as text files

We'll crunch numbers and send everyone
anonymized averages back on a weekly basis



The image shows a screenshot of a Notepad window titled "example-eval-template.txt - Notepad". The window has a menu bar with "File", "Edit", "Format", "View", and "Help". The text inside the window is as follows:

```
TEAM: Example Team
MY NETID: RZU946

SCORES:
* RZU946 Robert Zubek: 100
* HAX000 Elite Haxxor: 150
* SLA999 Slacker Dude: 50
```

Homework assignments summary

1. Project status update
 - One per group
2. Teammate assessment
 - Individual, everybody
3. Playtesting scores
 - Individual, 25% of the class each week

Due EOD Sunday each week

Questions on playtests or reports?

Homework

Submit a high level project plan (details on next slide)

This is **not a binding estimate!**

It's a planning exercise for you,
so that you know if your game is over-scoped.



Homework

Submit a high level project plan (one per team)

- Assume weekly sprints starting last Monday (presentation day)
- Describe the team's goals for each sprint
 - What do you want to have done at the end of each week?
 - When will you have something playable to iterate on?
- Verify: do these steps add up to what you described in your project proposal?
 - If not, that's okay. Describe how you're revising your project.

Due EOD Sunday



Afterwards: weekly status updates

These will start next week

One pager that states:

1. Result of last sprint
2. Dates of last sprint standups
3. Playtesting results (triaged feedback list)
4. Task items for next sprint with effort estimates

We'll be doing these every week.

The end! Questions?

1. Project planning
2. Weekly sprints
3. Playtesting and reports

End of class: Homework!