How to Write Great Design Documents

Damion Schubert
Bioware Austin

www.zenofdesign.com

About Me

- MMO Designer for 10 years
  - Lead Designer for most of them
- Used to working with very complex systems
- Grew to appreciate good documentation processes.
- Found being the ‘doc guy’ very good for my career
- Still learning about how to do it right

What I hear

- “Design documentation is a waste of time”
- “No one reads design docs.”
- “My programmers find reviewing design documents is a waste of time.”

This is probably a testament to your documentation, not to documentation in general.

The Harsh Truth

- All designers should want to share their ideas
- All programmers (and other team members) should want to know what they are building.

- That being said, most design documentation I run across isn’t very good, and ignores the iterative process of finding fun.

This Talk

- The Goals
- The Problem
- Generating The Idea
- Function
- Form
- The Process
- The Scrum

Presentation Focus:

- Systems Design Document

Talk is not about:

- Executive Summaries/Vision Documents
- Design Overview Documents/DDRs
- Test Plans

Goals of Good Docs

- Capture design consensus
- Primary vision conduit between departments
- Aid in scheduling
- Offer focus
- Give visibility to future dependencies and design conflicts

Why is good design documentation so rare?

- They deal with complex, interconnected systems.
- Designers tend to overdesign.
  - Systems take less time to design than to build.
  - “Big Book of Stupid”
- They don’t embrace iteration.
- They are rarely kept up to date as the project progresses.

What other devs say:

“ Just give me something that’s short, targetted, and up-to-date.”

“Short and accurate, easy to find the code bits”.

“ I just want a bullet list of things to do.”

The Idea

1. Always start with a Kickoff Meeting

- Get with your Lead Designer, and ask these three questions:
  - What are the goals of this system?
  - What are the questions this document should answer?
  - How complex can this system be?

2. Identify the goals

- Do this with your lead designer in the kickoff meeting
  - Justify the system
  - Help decide fencepost issues
- Example: the following two goals are worthy, but contradictory, unless the design plans for it up front.
  - “Crafting is a sideline activity, to fill downtime, and can be done on the field.”
  - “Dedicated crafters can own their own forges and blacksmith shops, and achieve fame and fortune serving other players.”

3. Define the Boundaries

- Do this with your lead designer in the kickoff meeting
  - Since all systems touch each other eventually, important to decide where a document ends.
  - Allows leads to schedule the documentation process.
  - Prevents jumping the gun, design ‘claim jumping’
  - Highlights phase 1 features

4. Determine Feature Ambition

- Do this with your lead designer in the kickoff meeting
  - **Token Representation.** We just want the bullet point on our box
  - **Competitive.** We want what the market leader has with minor tweaks, but we don't want to be too risky.
  - **Alternative.** Nothing too big, but definitely different from our competitor.
  - **Innovative.** This feature will crush opponents, and we will hear the lamentations of their women.

5. Research.

- Look at previous games.
  - Especially what didn’t work!
- Look at non-game subject matter information
  - If you’re making a cooking game, watch some Iron Chef!
- Talk to subject matter experts
  - Even on other teams and/or at other companies!

6. Brainstorm

- Many brainstorming techniques, all sharing the same general principles
  - Multiple (6-8) people
  - Encourage out of the box thinking
  - No idea is a bad idea
  - Identify the ideas that resonate.

7. Cull the Weak

- **Now**, some ideas are bad ideas.
- Ideas to cut
  - Unbuildable
  - Over-ambitious
  - Conflicting with goals, or with each other
  - Just plain weird ideas
- Use kickoff meeting results as your razor
- Might want to save the intriguing ones in a ‘morgue’ somewhere.
- Begin prioritizing the Darwinian survivors.

8. Keep it simple

- At least initially
- Focus on the essentials, emphasize getting something playtestable as fast as possible.
- Designs should become complex based upon feedback and design iteration.

If you are afraid of what people will say when they read your design document, it is probably either too complex or too weird.

9. Embrace Iteration

- **No game design survives contact with reality unscathed**.
- Don’t get emotionally attached to any aspect of the design.
- Prepare to iterate on designs once they are coded

Function

1. Know your Target

- People interested in a design doc:
  - **Design team.** To achieve design consensus.
  - **Programming team.** To build the game.
  - **Producers.** To schedule and go get money.
  - **QA.** To build test plans.
  - **External partners.** To reach quota of annoying demands.

1. Know your Target

- Programmers are the most important target.
  - It's how the game gets made.
  - Often, other documents are more useful for other audiences.
- Ask them what they want
  - If they say to ignore one of my rules, do it!

2. Keep it Short

- Easier to read
- Easier to write
- Easier to maintain
- Easier to handle politically
- Less likely to be contradictory
- More likely to be simple designs

2. Keep it Short

- Kill the fluff
- Kill empty sections
- Kill ‘cut and paste’ stuff
- Kill unnecessary text of obvious systems

2. Keep it Short

- Remember:
  - Programmers almost always want a short bullet list

2. Keep It short

Too Long

- **Guild Invitation Confirmation UI.** Players get a Confirmation UI when creating a guild. This asks “Do you really want to join this guild?” and has an ‘ok’ button and a ‘cancel’ button.
  - **OK Button.** The confirmation UI has an OK button, which confirms the transaction.
  - **Cancel.** The confirmation button has a cancel button, which prevents the guild from being formed.
  - **Close button.** There is an ‘X’ button in the upper right hand corner of the UI, which is identical to the cancel button.
  - **Esc.** Pressing escape will cancel the transaction, and performs identically to hitting the cancel button.

2. Keep It short

Better

- **Guild Invitation Confirmation UI.** Players get a confirmation dialogue when invited to a guild (see CommonDialogs.doc).

2. Keep It short

Who cares?

- **Crafting Tithe.** Hephaestus, the god of the forge, has instituted his will upon the craftsmen of Athens, and all are humbled by his greatness. As such, any players who wish to craft any items must pay a tithe to the temple of Hephaestus to earn his favor, unless he has found an item like the Hammer of the Gods, which allows the player to bypass these tithes.

2. Keep It short

Better

- **Crafting Tithe.** Players who craft items must pay a tithe to the local temple when crafting.
  - **Bypassing tithes.** Certain tools allow the player to bypass the tithe.

3. Prioritize the Design

- Give the features a priority, break them into phases
- Be sure document clearly separates out later phase features.

3. Prioritize the Design

Wrong!

- Players can equip items on the inventory screen.
- Equipped items change their combat stats.
- Player equipment is visible when worn.
- Player equipment may be enchanted with special effects
- Players may have their guild insignia drawn on their player shields.

3. Prioritize the Design

Still not great

- (Phase One) Players can equip items on the inventory screen.
- (Phase One) Equipped items change their combat stats.
- (Phase Two) Player equipment is visible when worn.
- (Phase Two) Player equipment may be enchanted with special effects
- (Phase Four) Players may have their guild insignia drawn on their player shields.

3. Prioritize the Design

Better

Basic Equipment **(Prototype)**

- Players can equip items on the inventory screen.
- Equipped items change their combat stats.

Advanced Equipment **(Phase 2)**

- Player equipment is visible when worn.
- Player equipment may be enchanted with special effects

Guild Insignia on Equipment **(Phase 4)**

- Players may have their guild insignia drawn on their player shields.

3. Prioritize the Design

- Phase 1: Prototype feature.
- Phase 2: Core feature.
- Phase 3: Must be in shipped product
- Phase 4: Wishlist, possibly expansion
- Phase 5: Yeah, right

4. Illustrate

- A picture is worth a thousand words.
- Tactics:
  - Screens of other games with similar features.
  - Visio diagrams
  - ‘Example’ text

4. Illustrate

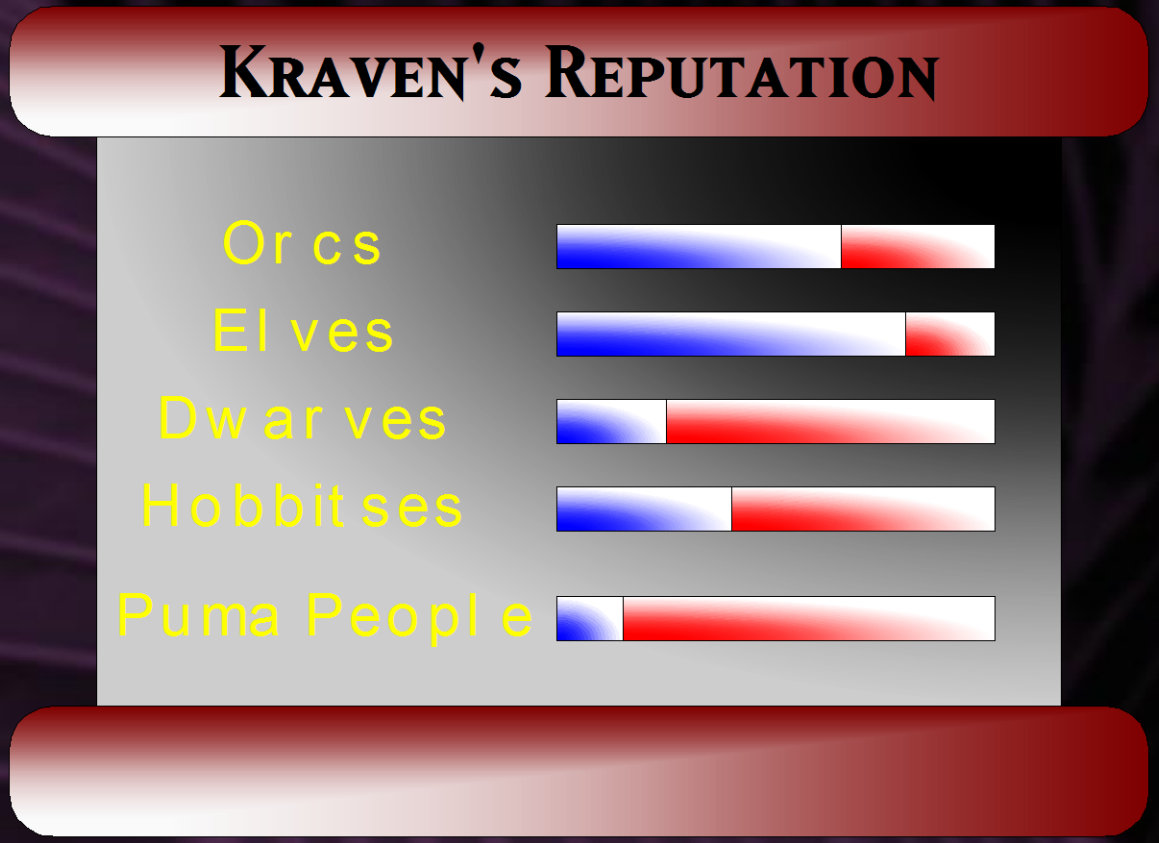
Example

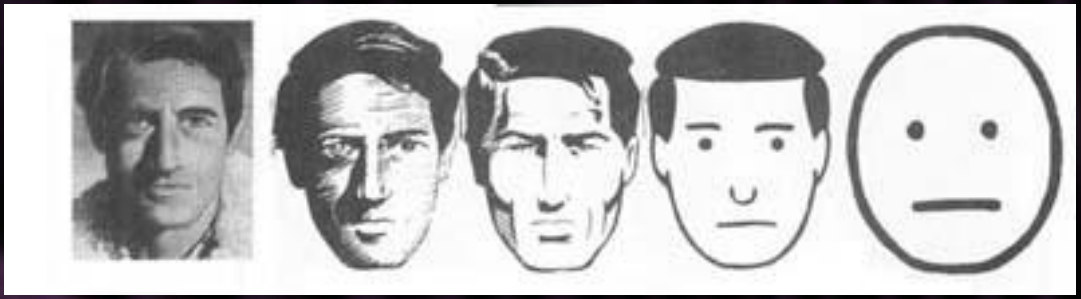
- Players can remove a skill in their skill tree by going to a special NPC (the ‘mindwiper’) and selecting that skill.
  - Removing a skill has a monetary cost in credits.
  - The player cannot remove a skill that is a prerequisite for another skill in his skill tree.

Joe Bob decides that he wants to unlearn Basic Psionics and Advanced Psionics, so he goes to a mindwiper. He tries to remove the Basic Psionics skill tree, but the transaction fails, as it is a prerequisite for Advanced Psionics. So Joe Bob unlearns Advanced Psionics and then Basic Psionics. In this case, both boxes are successfully removed.

4. Illustrate

What's wrong with this?

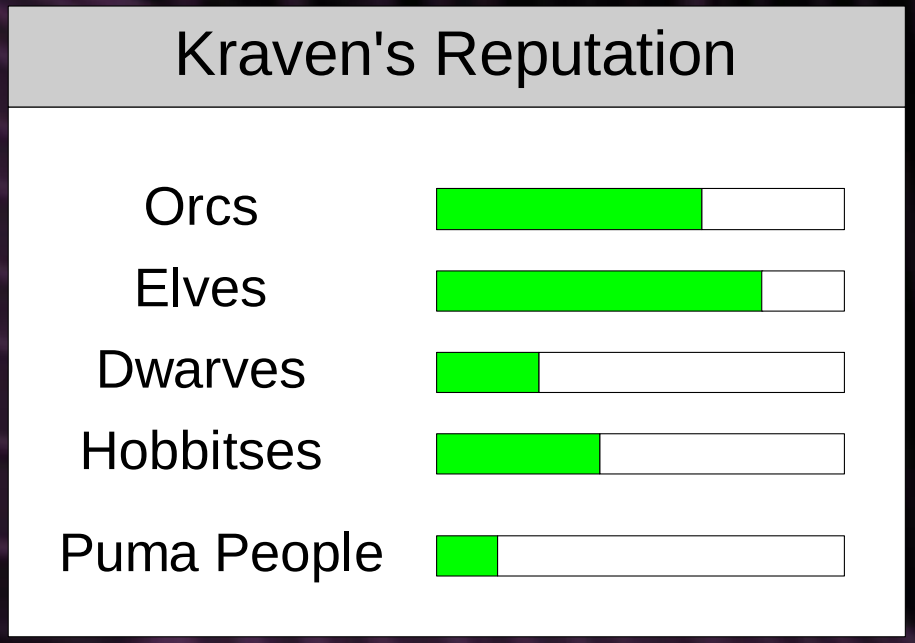




- The more abstract a picture is, the easier it is for a reader to project his own viewpoint.
- You want to give your UI artist something that is functionally accurate, but let HIM decide the aesthetics.

5. Don’t tell others How to do their jobs

Better, believe it or not!



5. Don’t tell others How to do their jobs

This is not your problem!

“Quests.doc”

- Quest Variables will be stored in a linked list of bitvectors on the character object.

5. Don’t tell others How to do their jobs

This is your problem. Let coders solve it.

“Quests.doc”

- **Memory considerations of quest variables.**
  - There will be approximately 2500 quests in the game.
  - Players may have 20 open quests at a time.
  - Players can make up to 10 decision points in one quest, the status of which must be stored until the quest is completed.
  - Players may find content later which is unlocked by quests they have already completed – the completion state (and outcome) of a quest must be stored.

6. Capture your Reasoning

- But compartmentalize it.

No!

- Players may not place items on the ground. This is to help reduce visual clutter and ensure that players may not be disruptive through the placement of hundreds of items.

6. Capture your Reasoning

- But compartmentalize it.

Much better!

- Players may not place items on the ground.

…

FAQ: Why can’t players place items on the ground?

This is to help reduce visual clutter and ensure that players may not be disruptive through the placement of hundreds of items.

6. Capture your Reasoning

- Capturing your reasoning is especially useful for longer projects, where the team may literally forget why they chose one side or the other.
- Capturing your reasoning, by extension, reduces the number of times contentious issues are reopened.

FORM

1. Separate Code from Content

Scary wall of bullet points!

- **Crafting tools.** Some crafting skills will require crafting tools to be used, or the player will get an error message saying he cannot use that skill.
  - **Blacksmith.** Using blacksmith skills requires a blacksmith hammer and tongs. Players may eventually find more advanced hammer and tongs, that give access to more crafting options.
  - **Tailor.** Being a tailor requires a loom.
  - **Alchemy.** Alchemy requires a test tube set. Players may eventually find more advanced test tubes, that give access to more crafting options.
  - **Sculpture.** Sculpture requires a hammer and chisel.

1. Separate Code from Content

Only two requirements – easy!

- **Crafting tools.** Some crafting skills will require crafting tools to be used, or the player will get an error message saying he cannot use that skill.
  - **Advanced tools.** Some crafting skills let the player craft more powerful items with more powerful tools.

Crafting Skills and Tools		
Skill	**Tools**	**Advanced Tools**
Smithing	Hammer and Tongs	Yes
Tailoring	Loom	
Alchemy	Test Tube Set	Yes
Sculpture	Hammer and Chisel	

1. Separate Code from Content

- Don’t make people hunt for the information they want.
- Separate content into appendices, or into tables.

2. Invest in a good Format

- Use a team template
- Change the font
- Use horizontal lines
- Use callout boxes for example
- Use bullet lists
- Don’t be a slave to your format

Viva la Difference

- This is the default Microsoft Powerpoint template
  - Not very good looking, is it?
  - Taking a little time to change out your fonts or add a watermark can have a huge impact on how professional your documents feel.

3. Use Clear Terminology

Don’t assume what your readers know!

- This spell has a high DPS, but also has a hate reduction component to reduce aggro in raids.
- There can only be six spawn agents per superchunk.

3. Use Clear Terminology

- Use plain english
- Avoid Wonky terms
- Avoid company-specific terms
- Use new terms consistently
- Consider a glossary

4. Kill Redundancy

- Duplication is the devil, leads to confusion, update errors.

Redundant Department of Redundancy !

"CombatStats.doc"

- Strength increases the player's damage by STRENGTH/2.
- Dexterity increases the player's accuracy by DEXTERITY/3
- Body odor reduces the player's chance to seduce NPCs by BODYODOR/2

"Items.doc"

- Strength increases the player's damage by STRENGTH/2.
- Dexterity increases the player's accuracy by DEXTERITY/3
- Body odor reduces the player's chance to seduce NPCs by BODYODOR/2

4. Kill Redundancy

- Duplication is the devil, leads to confusion.

Make one doc the owner, point others to it.

"CombatStats.doc"

- Strength increases the player's damage by STRENGTH/2.
- Dexterity increases the player's accuracy by DEXTERITY/3
- Body odor reduces the player's chance to seduce NPCs by BODYODOR/2

"Items.doc"

- Enchantments on an item can increase the players stats when worn. See CombatStats.doc for more details.

5. No Weak language

No!

- Players might be able to woo NPCs of the opposite sex.
- In the future, we may add the functionality to increase your chances to woo women by playing sappy love songs.
- If this is implemented, maybe players can write their own love songs.

5. No Weak Language

Better!

Romancing NPCs (Prototype)

- Players can attempt to romance NPCs of the opposite sex by dialogue options
- Players can also attempt to romance NPCs of the opposite sex by serenading them with songs they've learned.

Advanced Romance (Phase Four)

- Players can craft their own songs for use in the romance system.

5. No Weak Language

- Use strong, declarative language
  - No ‘maybe’, ‘could’, ‘might’
  - Even avoid ‘may’.
- Don’t be ambiguous
- Don’t say ‘we’
- Choose a direction
- Move ‘maybe’ features to later phases.

The Process

1. Enforce the Kickoff Meeting

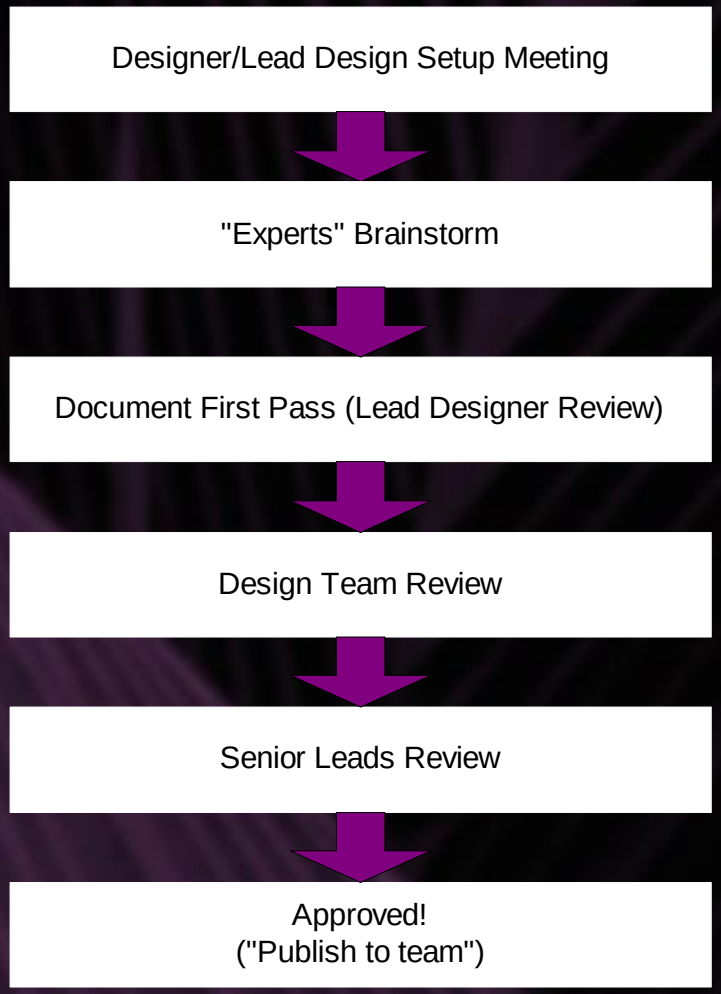
- Three questions:
  - What are the goals?
  - What are the boundaries of this design document?
  - What is the acceptable ambition level for this feature?
- Goal is to give designers autonomy in a well-defined way – to define the ‘boundaries of the box’.

2. Embrace Iterative Design

- Design the next immediate phase to fine-tooth detail
- Design other phases to man-month degree`
- Don’t emotionally invest in far-off features
- Revisit documentation as the design shifts and iterates.

3. Design Documentation is a collaborative Process

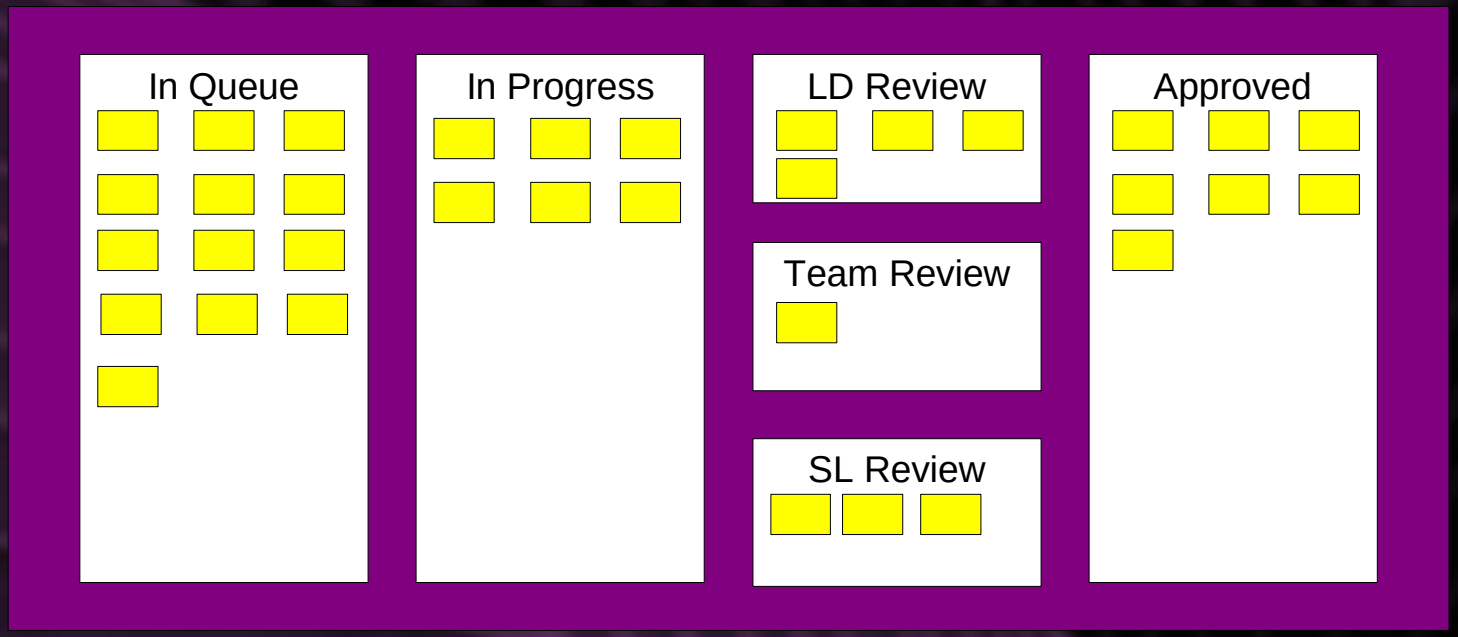
Design documents written in a vacuum almost never survive ‘contact with the enemy’.



4. Have an Approval Process

- Should telescope out
  - Lead Designer Approval First
  - Design Team Approval Next
  - Senior Leads/Cross-Team Approval Next
- This approach allows the design team to speak with one voice about a finished design.
- Is always tough to get up and running, but usually accelerates once teammates find value.

5. Have a visual Method of Tracking Progress



(I like using post-it notes)

6. Have a change Process

- Designs will shift as the game iterates. A process is necessary to ensure that design changes are disseminated to decision makers on the team.
- The Lead Designer can usually act as the arbiter of when a change needs to be approved by the Senior Leads.

7. Make it Searchable

- Design docs will only be used as a reference if the user can find what he needs.
- Possible means:
  - Wiki
  - Desktop Search
  - Design Bibles

8. Automate what you can

- Advantages of Documentation Automation:
  - Accuracy, even postscriptively
  - Searchable
  - Easy to add auditing and reports

9. Be Prepared to drop Trou

- You never know when external people will want to see your design documentation.
- Documentation should be ready to go on half-a-day's notice.
- Design team conflicts should not be visible in design documentation, or should be easy to excise.

You need to discuss with your producers what you will **never** show.

10. Consider expiration Dates.

- A stray idea:
  - All documents ‘expire’ after 6 months, after which designers have to go back and verify that said documentation is still valid
    - Check for changes in design
    - Check for changes in priority
    - Identify and signify aspects of design that have been implemented.
- Harder to do with a lot of documentation

Does your documentation need to stay up to date?

- Depends on your reality.
  - Yes for large projects
  - Yes for games with a long life (live components or expansions) and teams with high expected turnover.
  - Yes if your external partners are especially annoying.

11. Occasionally audit the process.

- Design documentation procedures must work for the team. If the team sees the documentation process as oppressive, the design documentation process will end up subverted.
- Never lose sight of your goals:
  - Short
  - Up-to-date
  - Programmer Friendly

Scrum and Documentation

Brief Overview

- Small multidisciplinary teams
- Manage their own fates
- Product Backlog of features and subfeatures to implement
  - Prioritized by ‘product owner’, usually producer or senior designer
  - Idea is team is always working on the most important thing first
- Scrum is designed to be ‘documentation light’
  - Some advocates even claim it should be docless

So what does this change?

- Surprisingly not much
  - Still need documentation to satisfy publishers and other external contract partners
  - Still need documentation for QA test plans
  - Still need the process that generates the ideas
- Focus changes
  - Iteration and post-implementation becomes a priority.
  - Non-programmer audiences become a priority.
- Most important change
  - Structure documentation to use user stories

Use user stories

- **I**ndependent – doesn't overlap other user stories
- **N**egotiable – details and implementation are less important than end user satisfaction.
- **V**aluable – written with the end user in mind.
- **E**stimatable – detailed enough for programmers to architect & schedule
- **S**mall – no more than a week.
- **T**estable – design and programming can agree when it's done.

Use user stories

- The player hears a sound effect when he gains a level.

Too small!

- The players can elect a new space ambassador .

Too boundless!

- When the player gains a level, he hears a sound effect, sees a particle effect, gains 3 attribute points, gains 5 skill points and gains access to a prestige class if he is level 10.

Too long!

Use user stories

We use 1 user story with subrequirements, equal to 2-5 programming days of work.

- **The player gains a level when he crosses the experience point threshold.**
  - The player hears a ‘ding’ sound effect.
  - The player sees a particle effect.
  - The player gains 5 attribute points to be spend on his stats.
  - The player gains 3 skill points to be spent on his skill tree.
  - If the player has reached level 10, he can apply his Prestige Class (see PrestigeClasses.doc)

Questions?