

FORMAL CODE REVIEWS

Philadelphia CFUG 6/5/2007

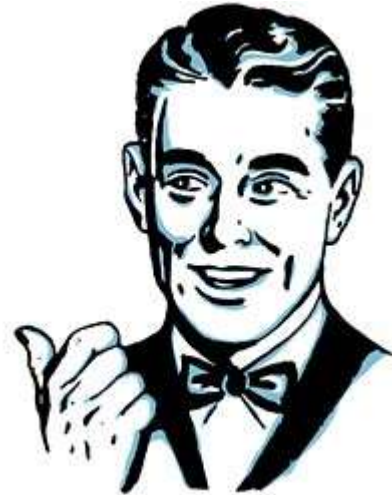
Speaker

- Terrence Ryan
 - ▣ Manager of Web Applications Environment at Wharton Computing
 - ▣ Write Aarrgghh!
(<http://www.numtopia.com.terry>)
 - ▣ Run servers, write backend code, write applications.
 - ▣ Sat through between 50-100 code reviews
 - ▣ First Review: 9/18/2000
 - ▣ Most Recent Review: 6/4/2007



What do you mean by Formal Review?

- A sit down meeting
- Everyone looks at the code
- Everyone talks about the code
- People bring up issues that need to be fixed before you can go ahead with application



What role do Formal Reviews Play

- Subtle errors of design
 - ▣ “When a user updates their profile, you don’t check that they are the same user they are trying to update. This makes you vulnerable to...”
- Code intention
 - ▣ You intend to return the cosine, but instead you are returning the sine.
- They can still find ideas caught by other layers of quality assurance.

Where do Reviews fit in?

- One of many quality tools
 - ▣ Design stage
 - ▣ Unit Tests
 - ▣ Informal Code Reviews
 - ▣ **Formal Code Reviews**
 - ▣ User Testing
 - ▣ Bug Tracking



Why Do Them?

Direct Consequences

- There's lot's of evidence that they are extremely effective at catching bugs.

- **Code Reviews: Just Do It**

(<http://www.codinghorror.com/blog/archives/000495.html>)

Indirect Consequences

- Accelerates learning
- Gives people a chance to talk code
- Reduce repetition of errors across the organization

How Do you Do a Review?



Unprepared!

- Organization has to take some steps first.
 - ▣ Define Goals
 - ▣ Define Standards (according to those goals)
 - ▣ Share Standards



Define Your Organizations Goals

- “Write better code” isn’t enough
 - ▣ Performance
 - ▣ Maintainability
 - ▣ Accessibility
 - ▣ Cross Platform
- You cannot write perfect applications. Choose what’s important.
- Collate goals from stakeholders
- Prioritize goals for organization

Example Stakeholder Goals

- Manager of a team with many small projects and high turnover:
 - ▣ Maintainability
 - ▣ Standards
 - ▣ Consistency
- System Administrator responsible for uptime
 - ▣ Performance
 - ▣ Security
 - ▣ Reliability

Example Organizational Goals

- Should be prioritized:
 1. Ensure code is secure and plays well with other applications on server
 2. Ensure code is maintainable
 3. Educate entire staff about common pitfalls.

Standards

- Standards are small bite-sized sentences that prescribe or prohibit techniques and operations.



Define Your Standards

- Should derive authority from your organizational goals
- Should be made clear if they pertain to the industry or the organization.
- Example:
 - ▣ Industry: Always var scope function local variables
 - ▣ Organization: Use mailserver1 for mail relaying.

Define Your Standards

Bad Standards

- ☐ Vague
- ☐ Absolutes
- ☐ All encompassing
- ☐ Subjective
- ☐ Stylistic

Good Standards

- ☐ Clear
- ☐ Flexible
- ☐ Concise
- ☐ Objective when possible

Standard Examples

Bad Standards

- ❑ Accessibility should be used
- ❑ Proper scoping should be used
- ❑ Do good and stuff with the database
- ❑ Never use Query of Queries

Good Standards

- ❑ All applications should pass standard Section 508 tests for accessibility (Industry)
- ❑ All application wide variables should be read directly from the application scope. (Organization)
- ❑ All tables should have a clustered index. (Industry)
- ❑ Avoid using Query of Query's for operations when you could more easily query the database. (Industry)

How do you Develop Standards?

Industry

- ☐ Blogs
- ☐ Forums
- ☐ CFUG meetings
- ☐ Journals

Organization

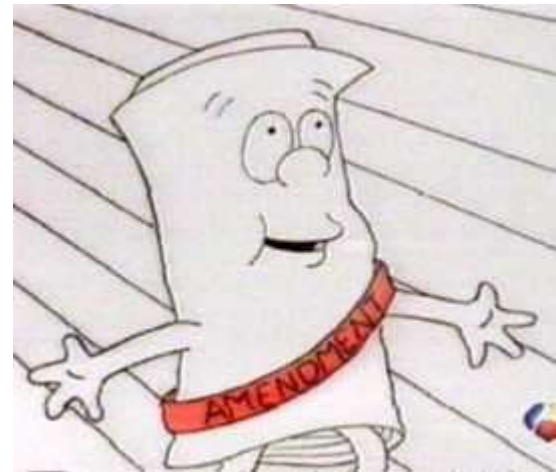
- ☐ Incidents
- ☐ Bugs
- ☐ Other code reviews

Share Your Goals and Standards

- Use some sort tool that allows for two way communication, and archive-ability:
 - ▣ Blog
 - ▣ Forum
 - ▣ Archived mailing list
- Allow for ongoing commentary
- Allow new programmers access to old knowledge

All powers not delegated to the standards are reserved for the goals...

- Goals inform code review notes too
- Just because something passes all standards doesn't mean it doesn't violate a goal or two.
- New standards evolve from uncaught goal violations



How Do you Do a Review?



Roles

- Reviewees
 - ▣ Explains application
 - ▣ Shares the content
 - ▣ Discusses solved problems.
- Reviewers
 - ▣ Criticizes code against organization standards and goals
 - ▣ Tries to make suggestions on how to fix
- Moderator - a reviewer who controls the meeting, not the content

Before the Review

Reviewee

- **Comment and format your code**
- Lock the code base
- Share the code ahead of time
 - ▣ Remove passwords
 - ▣ Remove any other sensitive material
- Give ample lead time. For a typical web application we give one week
- Schedule late in the day (No one reads the code until the day of anyway)

Reviewer

- **Read the code before the review**
- Have specific criticism ready to be discussed
- Cite file names and line numbers specifically
- If user does something consistently throughout, have at least one citation

During the Review



Reviewee

- ❑ Shut up
- ❑ Listen to the entire criticism
- ❑ Deliver defense in terms of the problem you were trying to solve.
- ❑ Your code is on trial, not you.



Reviewer

- ❑ Criticize the code, not the developer
- ❑ Before declaring a bit of code wrong, ask why it was done the way it was
- ❑ Remember this is your colleague, and they will be reviewing you in the future.



Moderator

- Keep review flowing
- Keep people on topic
- Break infinite loops



Reviews are not Technically Challenging

- Real problems are interpersonal
- Watch for:
 - ▣ Personal instead of code criticism
 - ▣ Axe Grinding
 - ▣ Stylistic criticism



Organization

- Break into sections
- I recommend by layer/technology:
 - ▣ Front Layer: HTML and CSS or Flex
 - ▣ Application Layer: ColdFusion
 - ▣ Persistence Layer: Database
- Also a good choice, by goal:
 - ▣ Security related issues
 - ▣ Performance related issues
 - ▣ Maintainability related issues

Severity

- Judge not just right or wrong, but how right and wrong.
- Example Security:
 - ▣ Passing a password over clear text –
 - Severe
 - Must change
 - ▣ Allowing a user to use dictionary words in a minimum length 9 character Alpha numeric password
 - Medium
 - Reconsider
 - ▣ Not keeping track of very old passwords to prevent reuse
 - Minor
 - Consider adding a feature on the next version.

Don't forget to Praise

- Make sure to notice something unique or elegant
- Acknowledge when a developer is trail blazing



Aftermath



Responsibilities

Reviewee

- Make the discussed changes

Reviewer

- Any new standards created?
 - ▣ Share them
- Any old standards forgotten?
 - ▣ Repeat them

Encouraging Participation

Mandated

- ❑ Get management buy-in
- ❑ Set some sort of dependency on getting a review done.
- ❑ People are motivated by the need to have others at their reviews

Unmandated

- ❑ Focus on improvement of process and developers.
- ❑ Point out that the developers who do this excel.
- ❑ Use this to get management buy in to get mandate for review.

Sinking in

- Best practices and standards will take time to sink in.
- Expect progress on a particular issue in about 4 to 5 reviews.



Improving Efficiency

- Automated code review tools:
 - ▣ Not really part of a formal review
 - ▣ Not that they are bad, just not really the same thing.
- Jupiter automates the process of conducting a review.
- Automate best practices

Resources

- Current Industry Standards

- ▣ House of Fusion

- <http://www.houseoffusion.com/>

- Cf-Talk Mailing List

- ▣ Full as a Goog

- <http://www.fullasagoog.com/>

- Code Review Tool

- ▣ Jupiter

- <http://csdl.ics.hawaii.edu/Tools/Jupiter/>

Conclusions

- ❑ Reviews do a lot of good things for your organization
- ❑ Reviews need to be driven by goals and standards
- ❑ Reviews can get emotional if you've got a team full of human beings
- ❑ If you stay on top of them, human issues can be managed

Questions

