Design or prototype first? [closed]

Asked 16 years, 1 month ago Modified 15 years, 4 months ago

Viewed 2k times

12

Closed. This question is opinion-based. It is not currently accepting answers.

Want to improve this question? Update the question so it can be answered with facts and citations by editing this post.

Closed 6 years ago.

Improve this question

When first approaching a project is best to step back and think through everything or just dive in and start coding and polish at a later date? Essentially, do you design first or try to rapidly prototype?

I have been burned by both methods, sometimes I try and think everything through but when I actually get down to the nitty gritty I encounter problems that I didn't take in consideration, and sometimes when I code first I end with code that needs to redone to fit in with a better overall design. Alot of my problems stem from inexperience, but any advice is welcome.

prototyping

Share

Improve this question

Follow

edited Aug 25, 2009 at 14:37

community wiki
3 revs
James McMahon

There is no real answer to this question, as it is highly subjective, so I am not going accept an answer.
– James McMahon Jan 15, 2009 at 4:35

13 Answers

Sorted by: Highest score (default)

30

Go incrementally and iteratively.

Design a bit, implement a bit.

Starting with a design you can suffer from a *tunnel effect* where you cannot have any real feedback before you actually implement something.

Starting without design, you can take decisions you'll regret.

The ideal situation is to be able to implement a very skeletal end-to-end version of your system that can be tested, and demonstrated to the customer.

Share Improve this answer

Follow

answered Nov 12, 2008 at 16:54

philant

35.7k • 11 • 73 • 113

5

It is always safer to design first, but this does not mean prototyping does not work. The real problem with prototyping is resisting the urge to keep the code you already wrote instead of throwing it away when the time comes to do the design.

Share Improve this answer

Follow

answered Nov 12, 2008 at 16:53

Maxam

4,053 • 2 • 26 • 25

Or the management urge to say "Well this works, why do you need more time?" – wonderchook Nov 12, 2008 at 17:27

Agreed! Similar to when you show off the prototype to the clients, and you can't make them understand that just because they see it doesn't mean it's working. – Maxam Nov 12, 2008 at 17:30

5

There is no silver bullet. It seems like design first is the preferred approach. But you will not be able to predict all complications that can arise while implementing your design. Some of them could potentially be show stoppers. Plus, if you're writing for a client, it's good to be able to show something just to make sure that you're on the same page.

At my workplace we do both - we do a rapid prototype, just to get feedback and get an idea of any potential problems. Then we do a formal design and formal implementation. In most cases we are able to salvage a lot of code from the prototyping stage. I like this approach, since we usually end up with clean, maintainable code.

Share Improve this answer

Follow

answered Nov 12, 2008 at 16:58

Filip Frącz

5,937 ● 11 ● 47 ● 67

4

See Gall's Law. The key is to iterate: design a little, implement a little, test a little, then repeat until you (or your customers) are satisfied. This is the essence of the new breed of "agile" methodologies.

Share Improve this answer

Follow

answered Nov 12, 2008 at 16:55

florin

14.3k ● 6 ● 49 ● 47

2

It depends.

Prototyping is most useful when the requirements or a solution aren't necessarily clear. As an example, I am doing a data warehousing project in an environment (large commercial insurance) where financial reconciliation is a big deal. This project has involved a large prototyping exercise to get a system that will

reconcile to the financials. As the business rules surrounding this were not well documented, the prototype was instrumental in exposing all of the corner cases.

In other cases, a design-first approach might be more appropriate. This is most applicable where requirements and a sensible solution architecture are reasonably obvious.

Share Improve this answer

Follow

answered Nov 12, 2008 at 16:57

ConcernedOfTunbridge Wells

66.5k ● 15 ● 148 ● 198

1

You must have some idea of a cohesive architecture before you start working. This is especially true of large scale systems.

Prototyping could be used for particular aspects of the design, e.g. presentation layer.

Share Improve this answer

Follow

answered Nov 12, 2008 at 16:54

Rob Wells

37k ● 13 ● 84 ● 147

1

I think it depends on what kind of business requirements you have up-front. If they are (relatively) detailed and complete, then I'd design based on those requirements. If you have barely anything to work with in the beginning,

then prototype out and show your customer what you got, to receive further requirement info.

Share Improve this answer

Follow

answered Nov 12, 2008 at 16:56

Kon

27.4k ● 12 ● 63 ● 87

1

You should develop using Agile Methodologies. Simply put, you design has you go. The team together with the product owner define a list of topics to develop, order them by importance, and split the development in iterations. Each iteration as features to be developed and on the start of the iteration is design each feature.

See more here.

Share Improve this answer

Follow

answered Nov 12, 2008 at 16:56

Bruno Shine

1,031 ● 1 ● 9 ● 17

+1 for not linking to your website in the answer ;-)
– Steven A. Lowe Nov 12, 2008 at 17:22

1

When first approaching a project, prototype. But don't prototype everything. Prototype one important thing (one "use case" if that means anything) and "turn the inner eye to follow its path" - keep an eye out for the practical problems you encounter in trying to get that one thing done.

Now that you have some idea what it takes to do an important thing, you can design from more than just first principles.

Of course, this assumes you're working in an environment where you can turn out prototypes at minimal cost to ongoing development efforts. But if you're working in such an environment, pepper your design discussions liberally with prototypes. With any luck you may get to keep some of them.

Share Improve this answer

Follow

answered Nov 12, 2008 at 17:12

Glazius

739 ● 7 ● 29

1

note that agile methods are not an excuse to avoid designing, they just encourage testing of the design more frequently, and in smaller increments

i like to sketch the design and break its elements down until reasonably sure that there are no obvious unknowns or risks; unknowns and risks are highlighted for 'spike' projects, with a time-box for determining feasibility and notes on possible alternatives if the preferred methods prove unworkable

once comfortable with the overall architecture, jump into the features bottom-up (or in priority order) to complete the design, write the initial tests, then implement

EDIT: note that the question "design or prototype first" is making a bad assumption, i.e. that it is possible to prototype without doing any design, which of course is not the case (unless you are using the million-monkeys methodology)

Share Improve this answer

Follow

edited Nov 12, 2008 at 19:01

answered Nov 12, 2008 at 17:26

Steven A. Lowe

61.1k ● 19 ● 135 ● 204

0

Design first, unless you're willing to take the risk of throwing out all the work put into your prototype when you find it can't do what you need it to do. At a minimum, you should make some high level designs for your project that can help you make some decisions about how you're going to build your prototype so that you will have a minimum of wasted effort.

Share Improve this answer

Follow

answered Nov 12, 2008 at 16:54

Elie

13.8k ● 25 ● 78 ● 128

0

If I know what I want to build, I just go right to design.

If I'm building something for a client, then I prototype to ease out more specific requirements from the users.

Share Improve this answer

Follow

answered Nov 12, 2008 at 17:00

John MacIntyre

13k ● 13 ● 69 ● 107

0

Maybe not an answer but a suggestion from my experience.

In most cases I'd be better off if I had started coding earlier. You can design until the cow comes home, but if the cows are on the horizon when you start coding, you might find your careful design hard to implement in time.

Share Improve this answer

Follow

answered Nov 12, 2008 at 17:06

John

16k ● 10 ● 74 ● 113