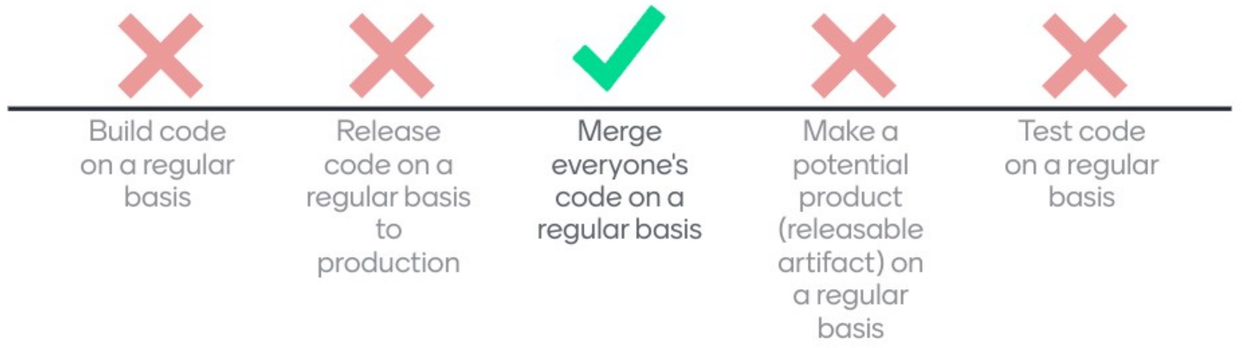
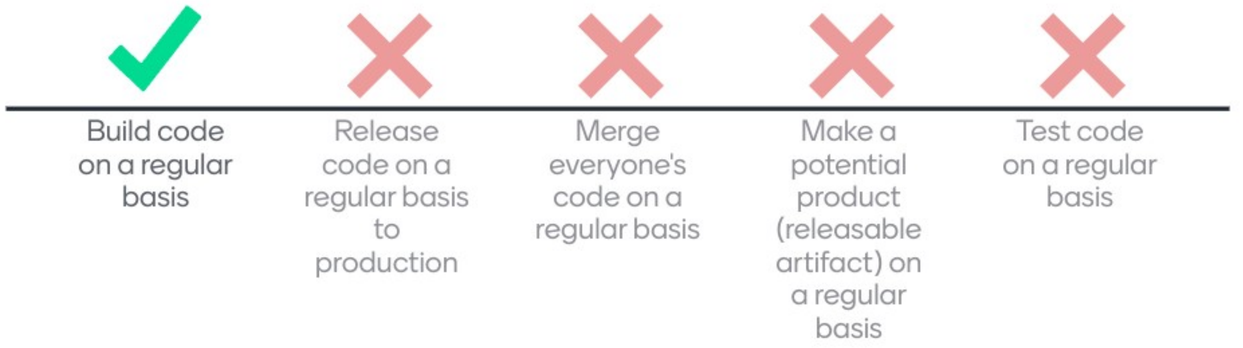
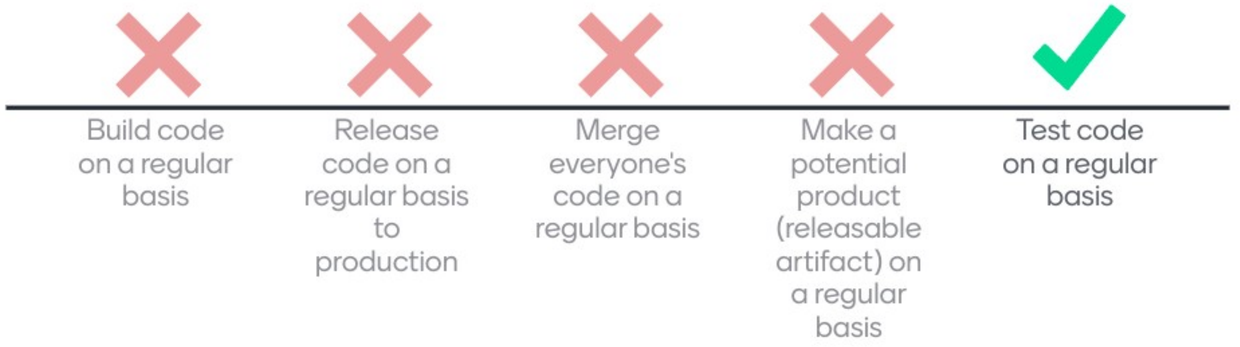
What is the first level of Continuous Integration?



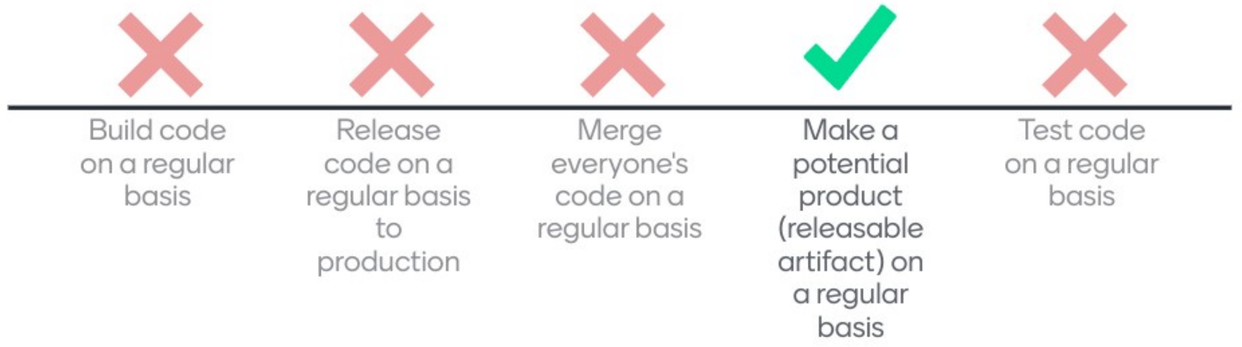
What is the second level of Continuous Integration?



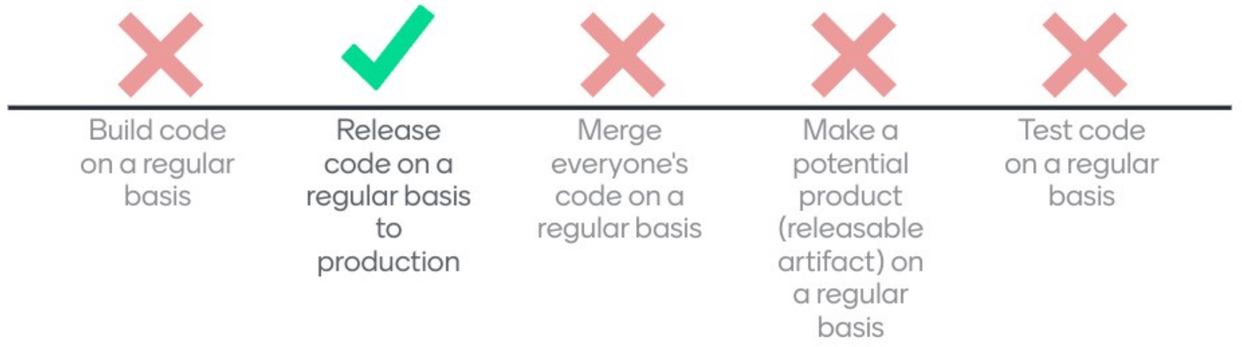
What is the third level of Continuous Integration?



What is the fourth level of Continuous Integration (Continuous Delivery)?



What is the last level of Continuous Integration (Continuous Deployment)?



Responding quickly to changing requirements?

Releasing working products often?

Sustainable pace?

Standup meeting

Scrum planning meeting

Scrum review meeting

Scrum retrospective meeting

Game must render at 60fps

must be able to mark a book "read"

must be able to mark a book "read"

API is documented using JavaDoc

User must be 18 years old to purchase

The software product must come in three tiers: home, professional, and enterprise

Customization?

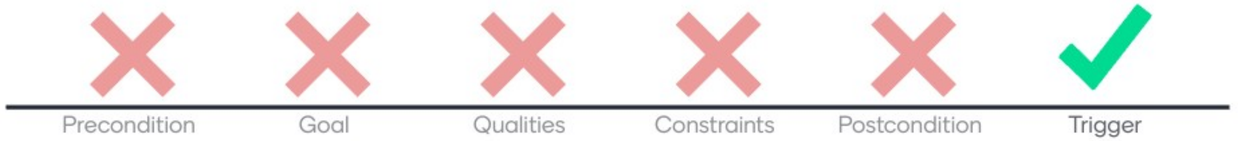
Guiding user through steps (wizards)?

Users are more likely to ...

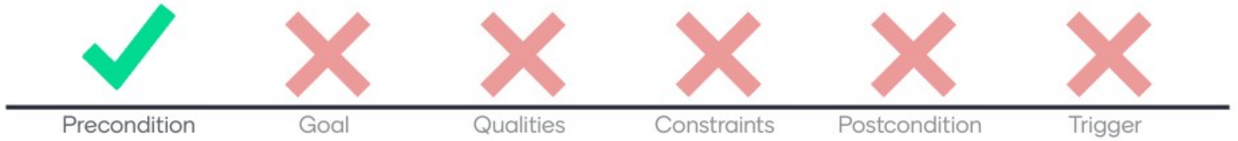
A user story with that can (and should) be split up into many smaller user stories...

We should AVOID putting what in user stories...

The first thing the actor does =

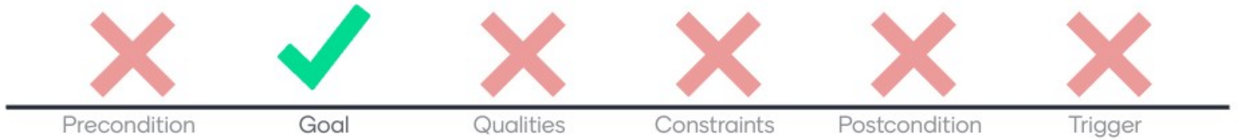


The things the actor knows & has

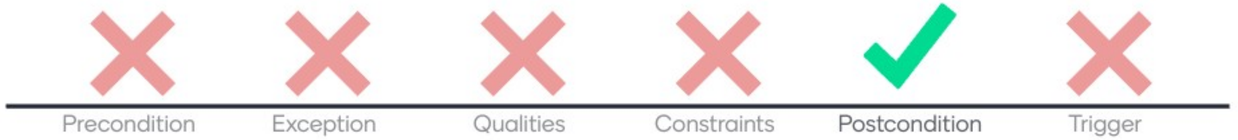


Usually has some non-functional requirements (NFRs)

What the actor is trying to do

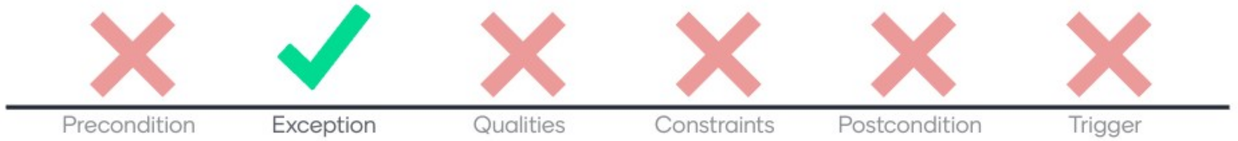


What the actor gets out of it if goal is met

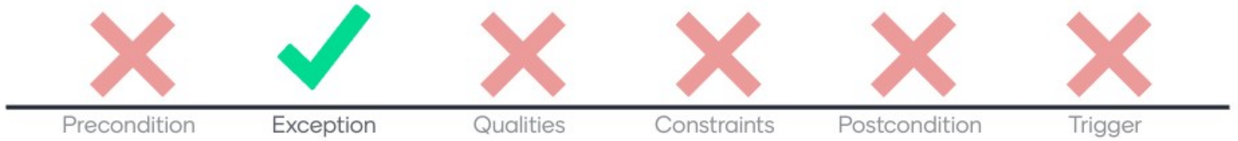


Restrictions on technical workings

Uses the same numberings as the basic flow



Uses the same numberings as the basic flow



User stories should be...

User stories should be...

User stories should be...

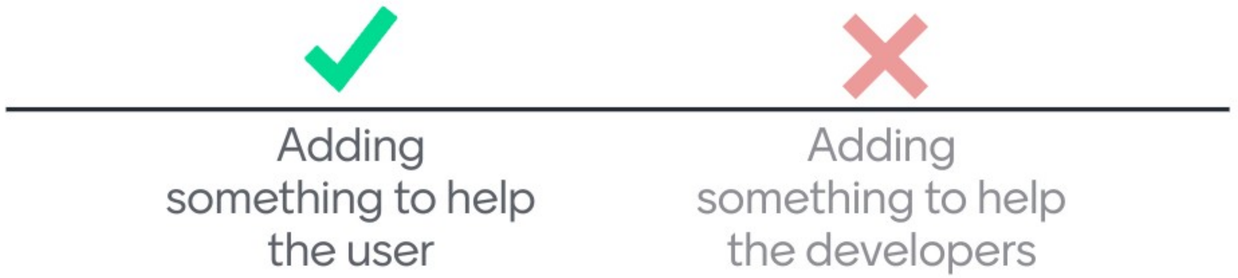
User stories should be...

User stories should be...

User stories should be...

User stories should be...

User stories should be...

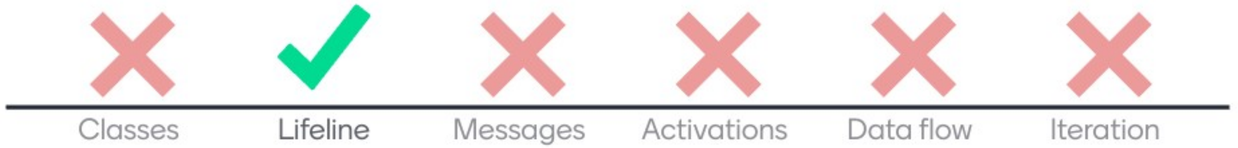


User stories should be...

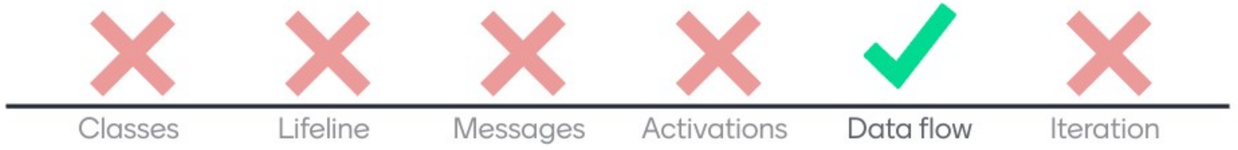
User stories should be...

In a UML Sequence Diagram, time goes...

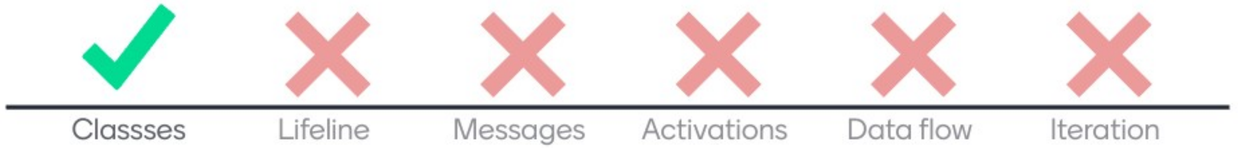
In a UML sequence diagram, vertical dashed lines...



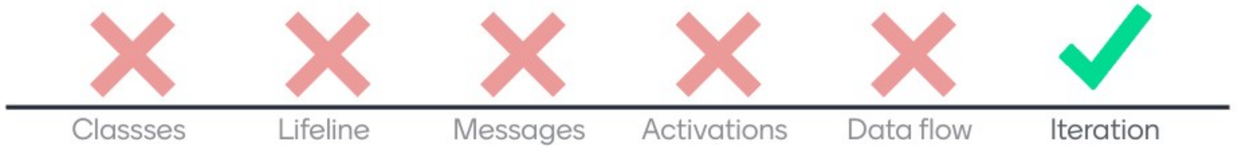
In a UML sequence diagram, horizontal dashed lines...



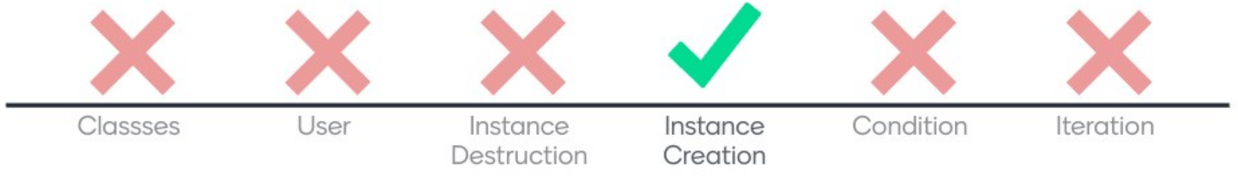
In a UML sequence diagram, columns represent...



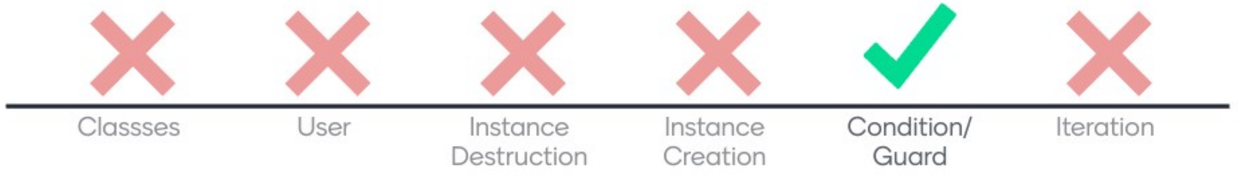
In a UML sequence diagram, * represents...



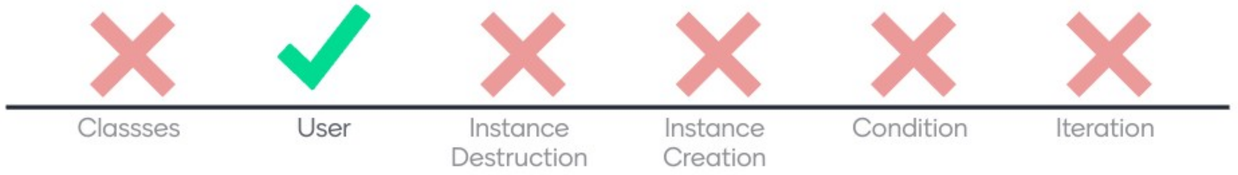
In a UML sequence an arrow pointing to the top of a column represents...



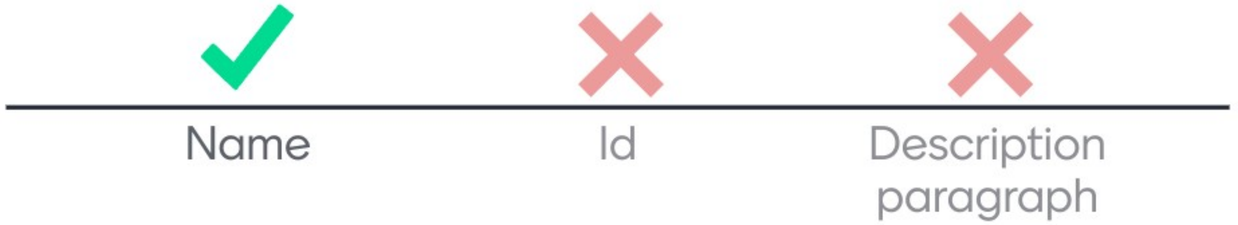
In a UML sequence square brackets [] represents...



In a UML sequence stick figure represents...



1st thing in a Use Case...



Next thing in a Use Case...

Next thing in a Use Case...

Next thing in a Use Case...

Next thing in a Use Case...

Next thing in a Use Case...

Next thing in a Use Case...

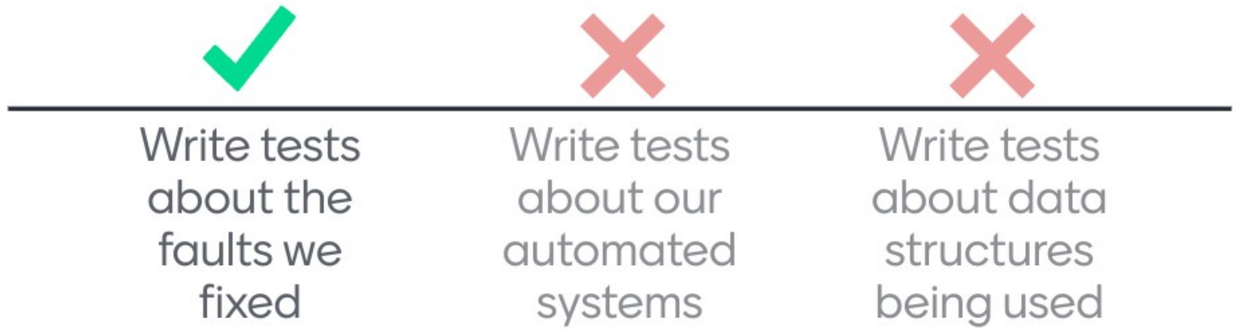
Next thing in a Use Case...

Next thing in a Use Case...

Next thing in a Use Case...

Black box testing

Regression testing

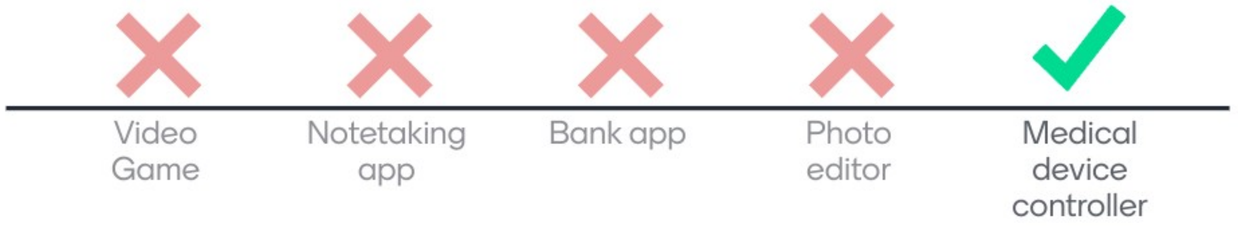


Equivalence classes: same class as 2+3=5 for a function add(int, int)

Mentimeter

Equivalence classes: same class as 57+(-27)=30 for a function add(int, int)

Which is likely to have the highest testing requirements?



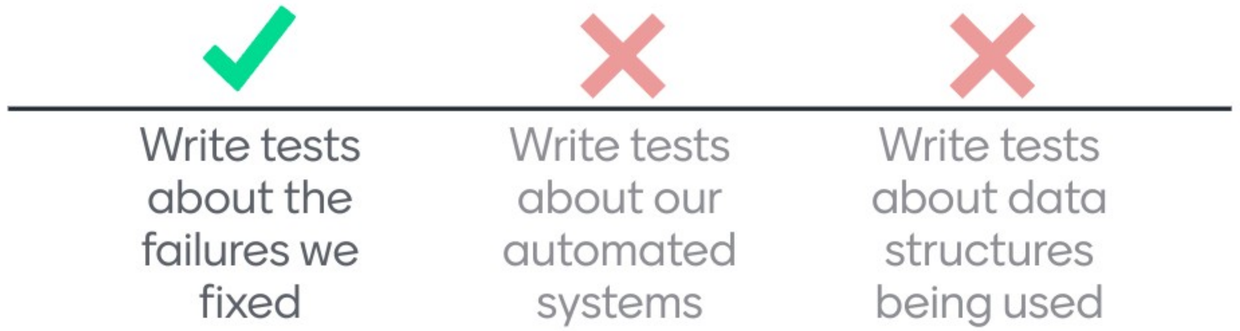
What's the main problem with of static analysis?

What's the main problem with test functions (JUnit)?

Black box testing

Mentimeter

Regression testing

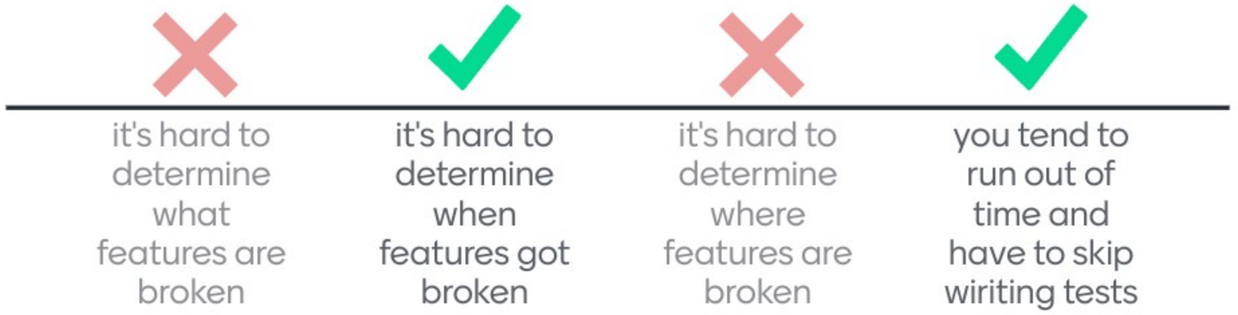


Top-down testing

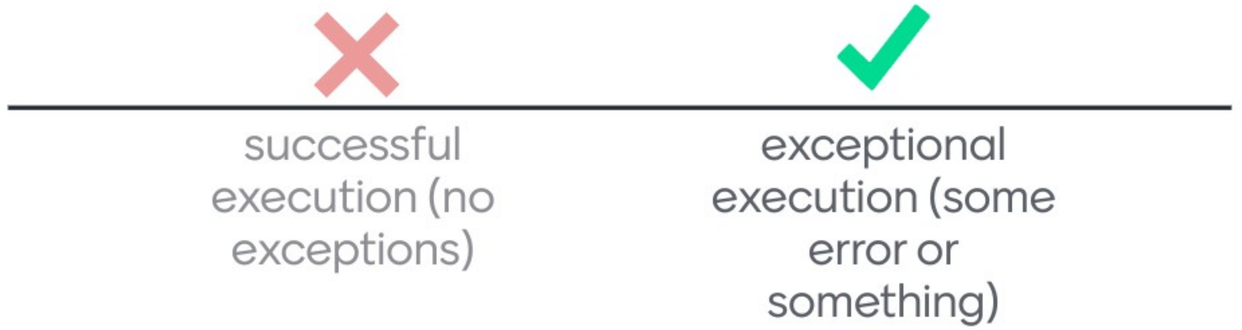
Mentimeter

Code that is hard to write tests for is also hard to...

Writing tests at the end of a project is bad because...



What to developers often forget to test



Testing a single class or method...

Testing multiple classes/methods working together...

Testing large chunks of the software from one end to the other...

What do we do with small bits of code that are repeated exactly?

What do we do with small bits of code that are repeated but always a little too different to extract to a common library?

It's important to write idiomatic code to

Design patterns let us

One class has a list of objects that it calls a method on every time something changes

Class that can only be instantiated once and there's only one way to access that instance

Each instance has a list of instances with the same superclass arranged in a tree

One class has a list of objects that it calls a method on every time something changes

Class that can only be instantiated once and there's only one way to access that instance

Class with instances that represent actions or changes

Each instance has a list of instances with the same superclass arranged in a tree

The template method for the template method pattern goes in the

The abstract methods for the template method pattern go in the

The override methods for the template method pattern go in the

"hooks" are similar to