O QUE VOCÊ FARIA COM 2 HORAS A MAIS POR SEMANA?

FASTLANE: AUTOMATIZANDO TUDO!

@_ant_one

PROBLEMS WE ARE GOING TO RESOLVE

- Someone click on the fixing issue button
- I don’t have the App Store provisioning profile
- I enter today and I just fixed a critical bug but nobody is here to publish the app
- I never remember how publish a new version of my component
- Need to open a web browser to open a pull request

FASTLANE IS THE SOLUTION

- Open source ruby projet create by Felix Kraus
- Set of tools to automatize your iOS development
- Part of Fabric
- Permit creation of lanes to automatize workflows

LET'S START WITH FASTLANE

```
$ gem install fastlane

$ fastlane init
```

A QUEST TO SOLVE CODE SIGNING PROBLEMS

FOREVER

match

match IS AWESOME!

- Keep everything inside a GIT repo
- No need to give access to dev portal
- Easy steps install match init
- Configurations : appstore | development | adhoc
- Single step on on the other computer or CI

match IS AWESOME!

```
$ match init

$ match appstore

$ match development
```

match IS AWESOME!

```
$ match development
```

THAT’S IT!

FIXING A 🐛 AND 🚀 A NEW APP VERSION ON YOUR FIRST DAY

FIXING A 🐛 AND 🚀 A NEW APP VERSION ON YOUR FIRST DAY

- Fixing the bug
- Increment build number
- Build for testing
- Running tests
- Build for release
- Sign with correct profile
- Upload to the App Store

WHAT'S INSIDE OUR FASTFILE

WHAT'S INSIDE OUR FASTFILE

```
fastlane_version "1.96.0"

default_platform :ios

platform :ios do
    before_all do

    end

    desc "Publish Beta version"
    lane :beta do
        increment_build_number
        scan
        match(type: "appstore")
        gym(scheme: "TDC-Fastlane")
        pilot
    end

    after_all do |lane|
        notification(message:"Finished lane: #{lane}")
    end

    error do |lane, exception|
        notification(subtitle: "Erro in lane: #{lane}", message:"Erro in lane: #{exception}")
    end

end
```

PUBLICANDO UM POD

PUBLICANDO UM POD

- install pods repo on local mac
- commit everything
- write changelog
- update pod version
- create a tag
- push the tag
- publish pod

```
lane :tag do |options|

   is_installed = sh "pod repo | grep private-repo | wc -l"
   is_installed = pod_repo_is_installed.to_i
   line_to_compare = 1

   if (pod_repo_is_installed <= line_to_compare)
      sh "pod repo add private-repo git@github.com:cpy/private-repo.git”
   end

  ensure_git_status_clean
  sh "git fetch --tags"
  last_tag = last_git_tag
  ver_numb = get_info_plist_value(path: plist,
                             key: 'CFBundleShortVersionString')

  if (Gem::Version.new(last_tag) >= Gem::Version.new(ver_numb))
     raise "There is already a tag with this number.".yellow
  else
    UI.success "it is ok, all good! 💪".green
  end

  changes = sh "git log #{last_tag}..HEAD --pretty=format:\"* %s - %an\""
  readme = File.read("../CHANGELOG.md")
  write = open("../CHANGELOG.md", 'w')
  write.write("##  ")
  write.write(version_number)
  write.write("\n")
  write.write(change_log)
  write.write("\n")
  write.write("\n")
  write.write(readme)
  write.close

  version_bump_podspec(path: podspec,
                            version_number: version_number)
  sh "git commit -am \"Publish pod version: #{version_number} \""
  add_git_tag(tag: version_number)
  push_git_tags
  sh "pod repo push private-repo .#{podspec} --sources='git@github.com:cpy/private-repo.git'"

end
```

AUTOMATIZED REALLY?

🤔

AUTOMATIZED REALLY?

“Why your computer should do hard work when another can do it for you?”

USE CONTINUOUS INTEGRATION

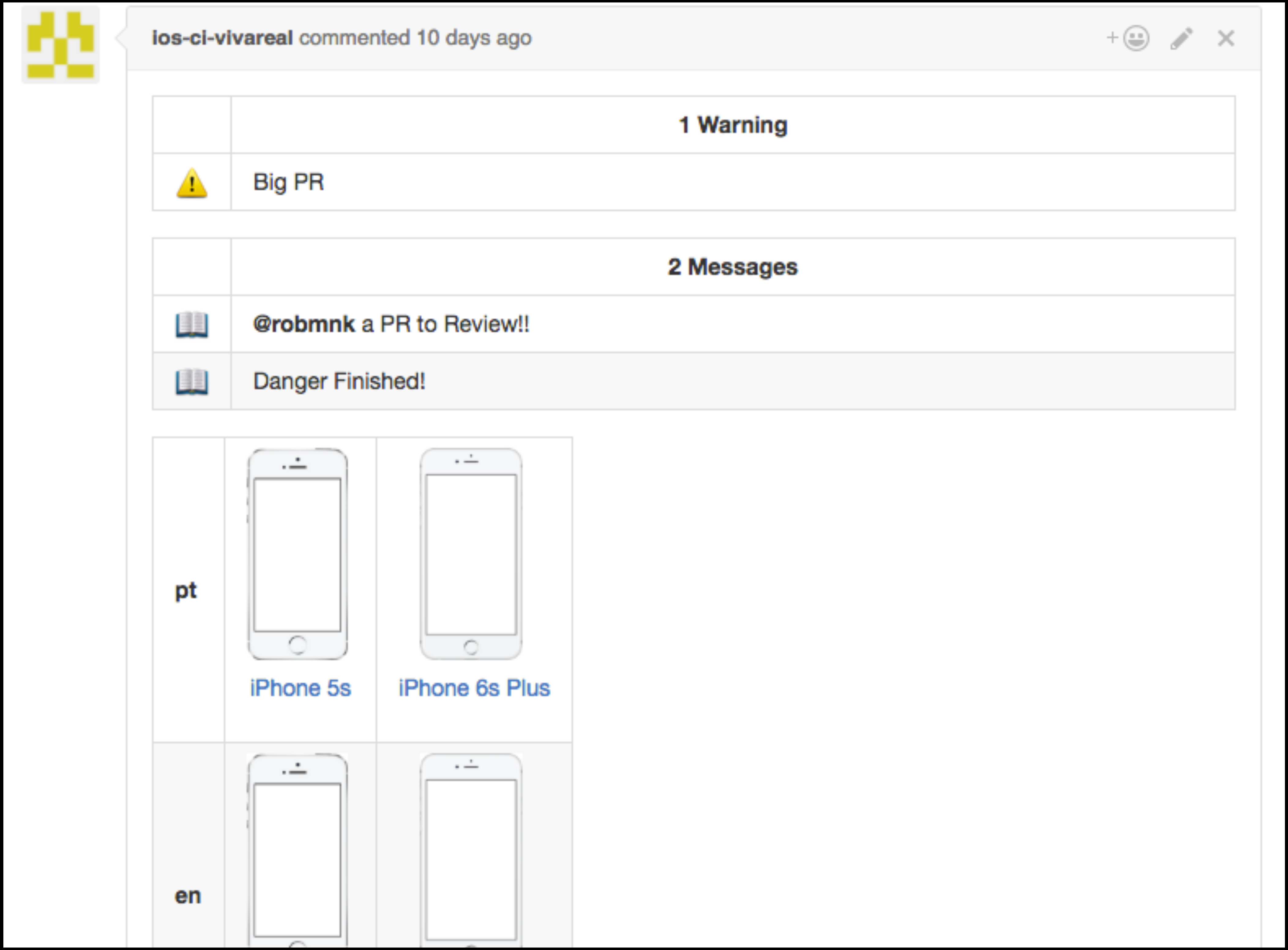
- Travis, Circle CI, Xcode Server, Jenkins
- Install fastlane if not available
- Override the default build configuration
- Run a lane as you will do on your mac

BETTER PULL REQUESTS

COMMON PULL REQUESTS PROBLEMS

- New feature but no changes on tests
- Poke your teammate that need to review your PR
- Not updated project number
- Pull request with short description
- Huge pull request

DANGER 🚫



ios-ci-vivareal commented 10 days ago

	1 Warning
⚠️	Big PR

	2 Messages
📖	**@robmnk** a PR to Review!!
📖	Danger Finished!

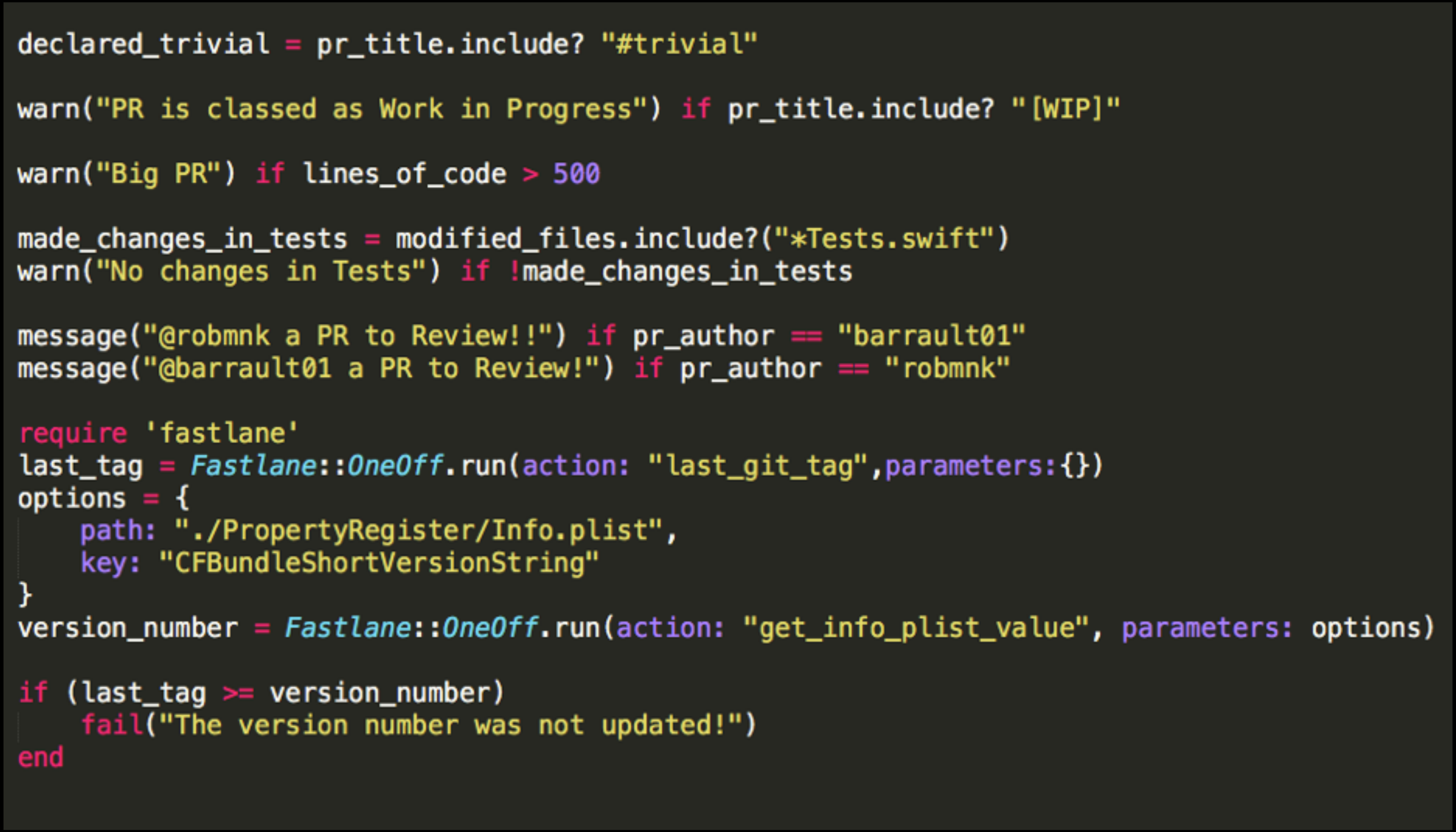
DANGER 🚫 BENEFITS 💰

- Files changes checks
- Pull Request size
- Pinging the other colleague
- Live Rendering in PR with apptize

BEGINNING DANGEROUS

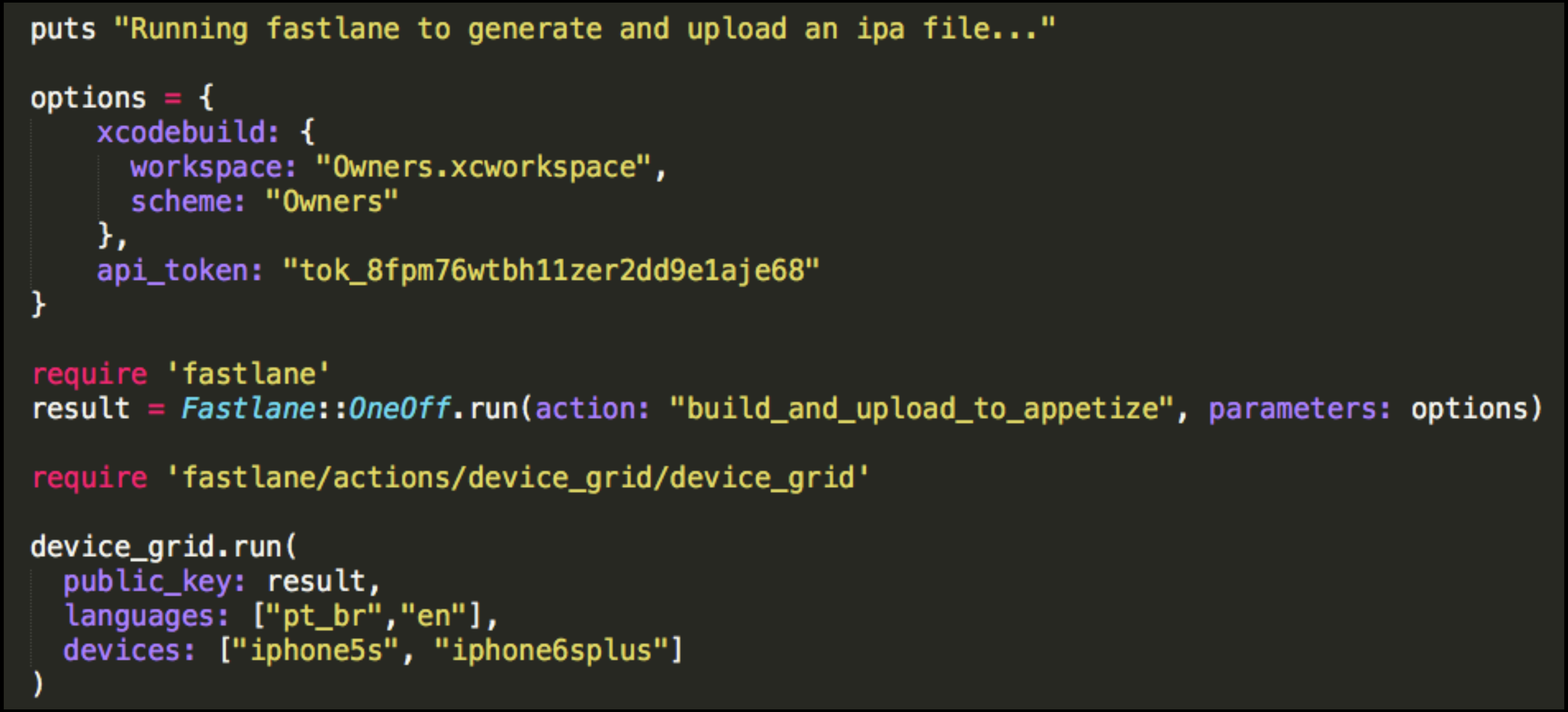
- Create a Gemfile
- add gem ‘danger’
- bundle install
- bundle exec danger init
- add to you CI file: bundle exec danger
- to test locally: bundle exec danger local

INSIDE A DANGERFILE



```ruby
declared_trivial = pr_title.include? "#trivial"

warn("PR is classed as Work in Progress") if pr_title.include? "[WIP]"

warn("Big PR") if lines_of_code > 500

made_changes_in_tests = modified_files.include?("*Tests.swift")
warn("No changes in Tests") if !made_changes_in_tests

message("@robmnk a PR to Review!!") if pr_author == "barrault01"
message("@barrault01 a PR to Review!") if pr_author == "robmnk"

require 'fastlane'
last_tag = Fastlane::OneOff.run(action: "last_git_tag",parameters:{})
options = {
    path: "./PropertyRegister/Info.plist",
    key: "CFBundleShortVersionString"
}
version_number = Fastlane::OneOff.run(action: "get_info_plist_value", parameters: options)

if (last_tag >= version_number)
    fail("The version number was not updated!")
end
```

LIVE RENDERING ON PR WITH APPTIZE



```ruby
puts "Running fastlane to generate and upload an ipa file..."

options = {
    xcodebuild: {
      workspace: "Owners.xcworkspace",
      scheme: "Owners"
    },
    api_token: "tok_8fpm76wtbh11zer2dd9e1aje68"
}

require 'fastlane'
result = Fastlane::OneOff.run(action: "build_and_upload_to_appetize", parameters: options)

require 'fastlane/actions/device_grid/device_grid'

device_grid.run(
  public_key: result,
  languages: ["pt_br","en"],
  devices: ["iphone5s", "iphone6splus"]
)
```

LET’S PLAY WITH

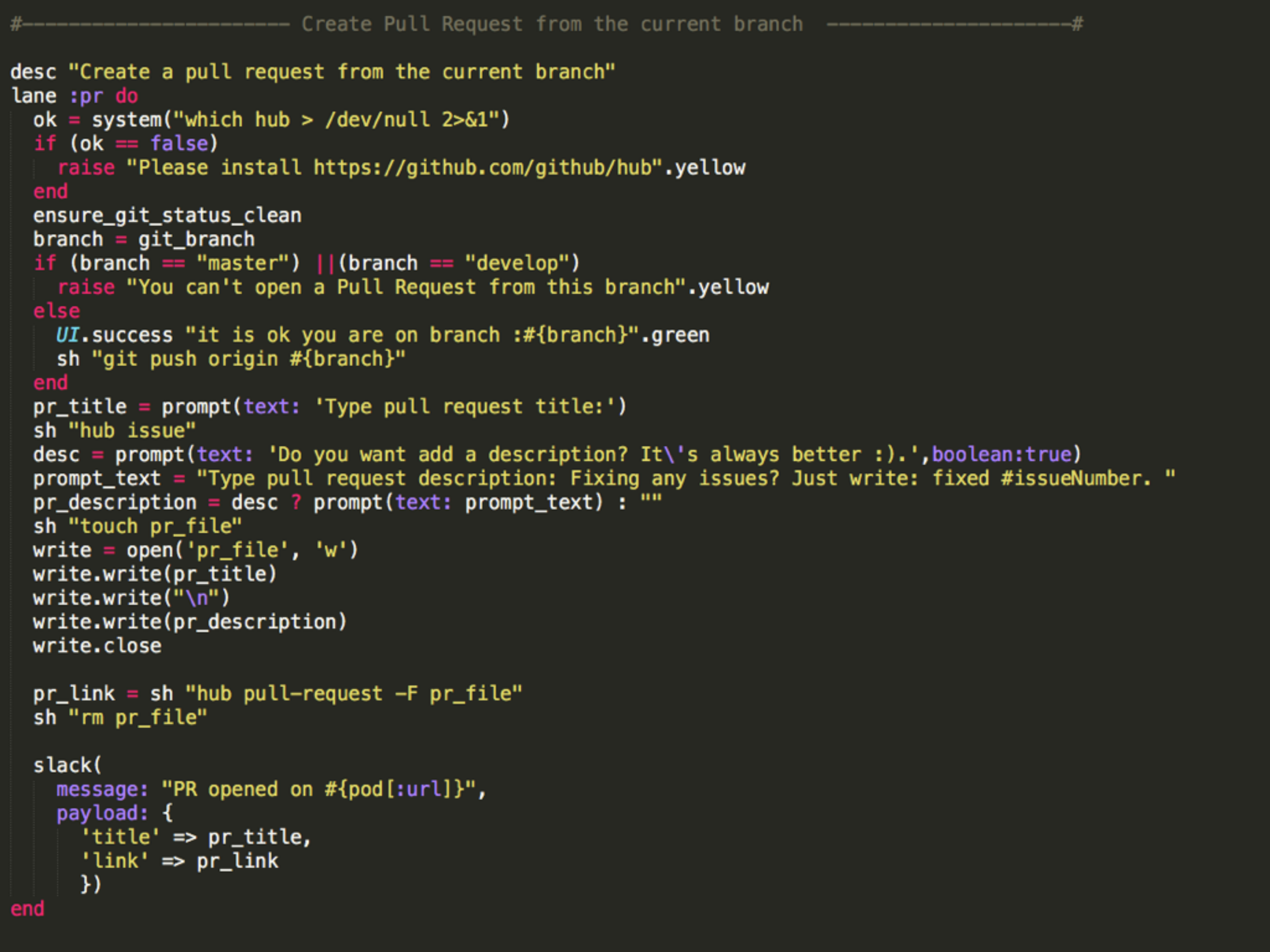
LET'S PLAY WITH

- Automate commits on bump
- Create a new tag based on project number
- Open Pull Request
- Listing open issues

LET'S PLAY WITH

```
desc "bump build version and commit"
lane :bump do
 increment_build_number
 build_number = lane_context[SharedValues::BUILD_NUMBER]
 sh "git commit -am \"Build version bump: #{build_number} [ci skip]\""
 build_number
end
```

```
desc "Tag the current repo status for App Store Release"
lane :tag_the_release do
  version_number = get_version_short_string
  time = Time.now
  sh "git tag -a #{version_number} -m \"Version #{version_number}
submitted to the App Store - #{time.strftime("%d %b %Y")}\""
end
```



```
#--------------------- Create Pull Request from the current branch  --------------------#

desc "Create a pull request from the current branch"
lane :pr do
  ok = system("which hub > /dev/null 2>&1")
  if (ok == false)
    raise "Please install https://github.com/github/hub".yellow
  end
  ensure_git_status_clean
  branch = git_branch
  if (branch == "master") ||(branch == "develop")
    raise "You can't open a Pull Request from this branch".yellow
  else
    UI.success "it is ok you are on branch :#{branch}".green
    sh "git push origin #{branch}"
  end
  pr_title = prompt(text: 'Type pull request title:')
  sh "hub issue"
  desc = prompt(text: 'Do you want add a description? It\'s always better :).',boolean:true)
  prompt_text = "Type pull request description: Fixing any issues? Just write: fixed #issueNumber. "
  pr_description = desc ? prompt(text: prompt_text) : ""
  sh "touch pr_file"
  write = open('pr_file', 'w')
  write.write(pr_title)
  write.write("\n")
  write.write(pr_description)
  write.close

  pr_link = sh "hub pull-request -F pr_file"
  sh "rm pr_file"

  slack(
    message: "PR opened on #{pod[:url]}",
    payload: {
      'title' => pr_title,
      'link' => pr_link
      })
end
```

MORE LANES

ADD NEW DEVICES

```
desc "Add new devices"
lane :device do

  device_name = prompt(text: 'Device name: ')
  device_UDID = prompt(text: 'Device UDID: ')
  register_devices(
        devices:{device_name => device_UDID})

end
```

NOBODY LIKES ITUNES CONNECT

```
desc “Send new version to apple"
lane :submit do

  deliver(submit_for_review:true,
          force:true,
          skip_metadata:true,
          skip_screenshots:true)

end
```

VISUAL TESTING USING SNAPSHOTS

```
desc "Create Snapshots"

lane :snap do
  snapshot(stop_after_first_error:false,
        reinstall_app:true,scheme:”UITests",
          devices:["iPhone 4s","iPad 2","iPad Pro”,
          "iPhone 5", "iPhone 6", "iPhone 6 Plus"])
  slack(message: "Time to check the pictures")
  clean_build_artifacts
end
```

WHERE TO GO NOW?

- https://github.com/fastlane/fastlane
- https://github.com/danger/danger
- https://fabric.io/blog/introducing-fastlane-plugins
- http://tech.vivareal.com
- https://github.com/barrault01

@FastlaneTools @KrauseFx

@_ant_one