Static and dynamic verification techniques -- Exercises

# Systems Engineering Laboratory 1

**Note:** Please treat these exercises also as professional work. For example, use meaningful commit messages.

**Note:** You should continue with your GitHub repository from the previous lab. If you did not participate in the previous lab let us know

Static techniques

# Code review

1. Clone your repository from the previous lab.
2. The readme of your repository contains the expected behavior of the system. However, changing the reference speed is not implemented yet. If the user changes the position of the joystick, the reference speed does not change (the function is written, but it is not called). Both members of the team should **create their own new branch** and **implement this feature** based on the readme. The periodic change of the reference speed can be implemented with for example timers, threads or with other constructs, it should be your decision.
3. After the implementation is complete, each member should commit their results (on their own branch), **open a pull request** (for their own branch) and request a review from the other team member.
4. Inspect the pull request of the other team member.
5. Go to the **Files changed** tab and **Start a review**.
6. Add comments to specific **source lines** identifying problems or suggesting changes.
7. After adding the comments you can submit your review with the button in the top right corner and select **Request changes**.
8. Your team member should make changes according to the review and add a new commit to the ongoing pull request.
9. If the commit solves the problem, then **Approve** the review.
10. After both reviews are complete, pick **one of two** implementations and **merge** that pull request into the master branch.

For more information see [Reviewing changes in pull requests](https://help.github.com/articles/reviewing-changes-in-pull-requests/).

# Sonarcloud

TODO (Code quality, Jacoco and code coverage, external dependency, new circleCI config)

# Heroku

1. [Heroku web] [Login](https://id.heroku.com/login) to Heroku dashboard using your already registered account.
2. [Heroku web] Create a new application. Remember the name you choose for your app.
3. [Heroku web] On the Deploy tab select Gitbug as deployment method.
4. [Heroku web] In the Connect to GitHub section select ‘FTSRG-ReteLab’ organization and search for your repository. Connect to your repository. (Megjegyzés: Admint kell adni a repora a résztvevőknek, mert webhookot kell állítaniuk.)
5. [Heroku web] In the Automatic deploys section tick Wait for CI to pass before deploy, then Enable Automatic Deploys.
6. [Heroku web] Look for your API key. Keep this sequence private. To find your key click your profile in the top right corner, select Account settings, Account tab, and in the API Key section click on Reveal. You will need this in the next exercise.
7. [local] Create a new job in your CircleCI config, which deploys your code **after** the build job is done. You will need your Heroku API key in order to successfully deploy your code.

Hint1: Use workflows

Hint2:

deploy:

docker:

- image: buildpack-deps:trusty

steps:

- checkout

- run:

name: Deploy Master to Heroku

command: |

git push https://heroku:$HEROKU\_API\_KEY@git.heroku.com/$HEROKU\_APP\_NAME.git master

1. [CircleCI web] **Important:** Do not include your Heroku API Key in your CircleCI config. Instead, use it as an Environment variable. For this, Login to CircleCI, in the Workflows tab click the gear icon next to your repository. In the Environment Variables tab Add your API key as ‘HEROKU\_API\_KEY’ and your app name (the name you created your Heroku app) as ‘HEROKU\_APP\_NAME’. You can reference these as $HEROKU\_API\_KEY and $HEROKU\_APP\_NAME in your CircleCI config.

(Megjegyzés: Ezt korábban, a sonarcloudos résznél meg kellett tenniük.

1. [local and CircleCI web] Commit and push your config to trigger build, check the web for results.
2. [Heroku web] Select your app in the dashboard, then click Open app in the top right corner.