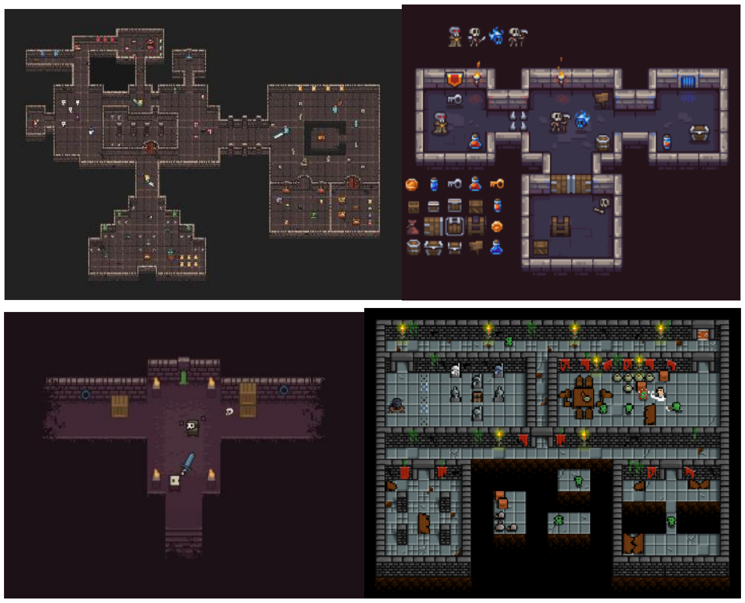
CS 2340 Sprint 1

New Game Configuration, Domain Modeling, and Use Cases

Due Date: See Canvas assignment

Background

The project theme this semester is a 2D Dungeon Crawler game. In this game, you will control a player's character and navigate through dungeon rooms with different layouts and enemies. These rooms will be from a fixed viewpoint, so it will not change based on player position. You can attack and destroy enemies, collect powerups, and reach exits that take you to the next room. The enemies will be able to attack the player, and the player will lose health. And if they run out of health, the game is over. To win, the player must reach the exit of every room. The player will be able to earn a high score on a leaderboard displayed at the end of the game.



Ref1, Ref2, Ref3, Ref4

Your project will be implemented across five sprints. The proceeding descriptions are basic ideas of what each sprint should cover. The sprint descriptions are subject to change, and requirements may be added or removed. *There may also be extra credit opportunities in certain sprints for certain extra features implemented.*

Purpose

There are two primary goals for this project. The first goal is to provide you with experience collaborating with a team to develop a product with specific requirements. The second goal is to increase your aptitude with several key software engineering principles and modern technologies alongside learning how to document your analysis and design with standard techniques. Over the course of the project, these concepts will be introduced to you with the expectation of incorporating them into your future sprints. With each following project sprint, testing will occur alongside development, and you will be responsible for writing unit tests to verify your implementation's functionality.

Tasks

For Sprint 1, you are asked to create and submit to Canvas multiple design deliverables along with demoing the functionality implemented. For the design deliverables, you will perform object-oriented analysis (OOA), produce a Domain model and Use Case Diagram, and submit evidence of how the team utilized Agile Development. For the implementation portion of this sprint, you will create four screens: a welcome screen, a player configuration screen, an initial game screen, and an ending screen. Other than the requirements outlined below, and that you must use Java (not Kotlin), the details of your implementation are up to you. Your application implementation and functionality will be graded during a demo, which will occur the week after sprints are due; see the class schedule for specific dates.

Design Deliverables

There are two design deliverables that are required in this sprint: a Domain Model and a Use Case Diagram. These are due **as a <u>combined single PDF</u> via Canvas submission (check course page for official date)**. For diagramming, we recommend using draw.io.

Domain Model

1. Identify and list at least ten potential nouns which could be used in your project. Some examples might be player, enemy, weapon, powerups, etc.
2. Identify classes and attributes:
    a. Classes (Game Objects) are nouns that require their own methods, attributes, and associations. Good examples of potential classes might include Player, Enemy, Weapon, etc.
    b. Attributes are descriptors of the identified classes (game objects). These are nouns that do not require a whole class to represent and describe a potential class. Good examples of potential attributes might include speed, direction, etc.
    c. List the identified classes and attributes on your submission PDF.
3. Draw a Domain model for the classes and attributes you brainstormed.
4. Connect each class within your Domain Model with at least one other class using associations.
    a. Example: Player "wields" Weapon
    b. Include multiplicities for each association, one on each side of the association.

Please explicitly categorize the nouns as either classes or attributes somewhere in your deliverable *in addition* to including the domain model. The submission of the Domain Model itself does **not** suffice for the inclusion of the listed and categorized nouns.

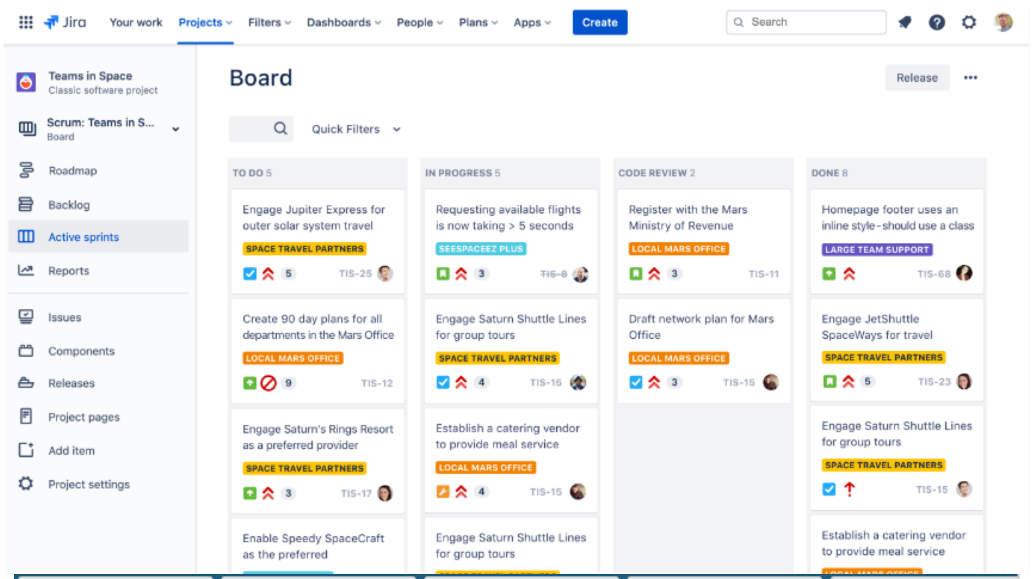
Use Case Diagram

1. Categorize the following actors as *Primary*, *Supporting*, or *Off-Stage*
    a. Player: attempts to control the game character and reach the goal tile
    b. Game Admin: uses the game's administrator screen to manage players
    c. Third-party database service: services the game uses to store its state (i.e., for high scores)
    d. Shareholder: provided financial backing to the software company that develops the game
    e. Contractor Company: localizes the game to other languages
    f. International Game Developers Association: provides documentation on implementation of common game features
2. Brainstorm one additional actor and categorize it like above

3. Draw a Use Case Diagram for the game application
    a. Include the player and at least three other actors
    b. Place Primary Actors on the **left** of the system boundary
    c. Place Supporting Actors on the **right** of the system boundary
    d. Place Off-Stage Actors near but **unconnected** to the system boundary and place a "Off-stage" label under them
    e. Include six or more Use Case Titles (functional requirements within the system boundary). These functional requirements should define the ways in which *Primary* and *Supporting* Actors may interact with the system

Agile Development

1. Submit a screenshot of the project management tool (Trello, Jira, Rally, etc.) used in Sprint 0.5 and Sprint 1. The screenshot should capture a clear view of the tool's board, showing columns/categories such as "Sprint Backlog", "In Progress", and "Done". Each task within these categories should have an assignee, indicating which team member is responsible for its completion. For best practices, it's advantageous if tasks also have associated "points" that represent the estimated effort required for completion. This serves as a visual indicator of the team's workflow and progression through the sprints. Some helpful examples are linked here:



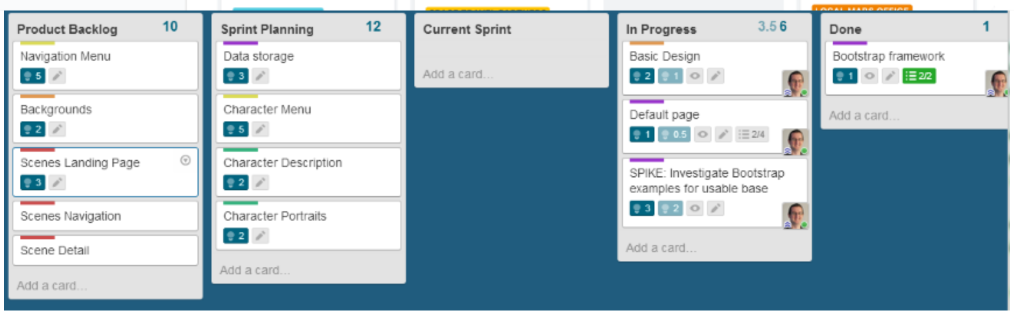
Board

TO DO 5
- Engage Jupiter Express for outer solar system travel — SPACE TRAVEL PARTNERS — 5 — TIS-25
- Create 90 day plans for all departments in the Mars Office — LOCAL MARS OFFICE — 9 — TIS-12
- Engage Saturn's Rings Resort as a preferred provider — SPACE TRAVEL PARTNERS — 3 — TIS-17
- Enable Speedy SpaceCraft as the preferred

IN PROGRESS 5
- Requesting available flights is now taking > 5 seconds — SEESPACEEZ PLUS — 3 — ~~TIS-8~~
- Engage Saturn Shuttle Lines for group tours — SPACE TRAVEL PARTNERS — 4 — TIS-16
- Establish a catering vendor to provide meal service — LOCAL MARS OFFICE — 4 — TIS-15
- Engage Saturn Shuttle Lines for group tours

CODE REVIEW 2
- Register with the Mars Ministry of Revenue — LOCAL MARS OFFICE — 3 — TIS-11
- Draft network plan for Mars Office — LOCAL MARS OFFICE — 3 — TIS-15

DONE 8
- Homepage footer uses an inline style - should use a class — LARGE TEAM SUPPORT — TIS-68
- Engage JetShuttle SpaceWays for travel — SPACE TRAVEL PARTNERS — 5 — TIS-23
- Engage Saturn Shuttle Lines for group tours — SPACE TRAVEL PARTNERS — TIS-15
- Establish a catering vendor to provide meal service



Product Backlog 10
- Navigation Menu — 5
- Backgrounds — 2
- Scenes Landing Page — 3
- Scenes Navigation
- Scene Detail
- Add a card...

Sprint Planning 12
- Data storage — 3
- Character Menu — 5
- Character Description — 2
- Character Portraits — 2
- Add a card...

Current Sprint
- Add a card...

In Progress 3.5 6
- Basic Design — 2 — 1
- Default page — 1 — 0.5 — 2/4
- SPIKE: Investigate Bootstrap examples for usable base — 3 — 2
- Add a card...

Done 1
- Bootstrap framework — 1 — 2/2
- Add a card...

Ref1, Ref2

2. Write a paragraph description of how the team utilized the project management tool in Sprint 0.5 and Sprint 1.
3. Write a paragraph description of how the team conducted scrum meetings for Sprint 1.

Project Setup

Sprint 1 through Sprint 5 all incorporate this semester-long project; therefore, it is imperative that you set up and utilize your repository in the correct manner.

1. The project must be implemented as an Android mobile application.
    a. The recommended IDE for development is Android Studio.
    b. There are several resources on getting started with Android Studio including Ed Discussion and the Android Studio video located in Media Gallery.
    c. Any external libraries must be cleared with the TAs by making a **public post on Ed Discussions** so that others may know of this library as well.
2. Your project must be hosted on the personal GitHub account you used for prior 2340 GitHub activities/lessons.
    a. Have a team member with GitHub Pro create a new repository for this application. Ensure that you follow this naming structure for your project: CS2340<Section Number>_Team<Team_Number>.
    b. **Do not use the repository you set up for Sprint 0.5.**
    c. **Ensure that your repository is private.**
    d. We highly recommend including a `.gitignore` file for your project. This allows you to exclude certain files that shouldn't be tracked (e.g., "`.DS_STORE`", "`local.properties`" files). For more information, see the following example gitignores:
        i. https://github.com/github/gitignore/blob/main/Android.gitignore
        ii. We also recommend looking at the Sprint 0 repositories to get an idea of where your Git top-level directory should generally be. You should be able to create a new project within the empty repository using Android Studio.
    e. Add the remaining teammates as collaborators to the repository.
3. Add the shared TA GitHub (CS-2340-Instructional-Team) as a collaborator to your repository.
4. Recall that this course expects you to create branches when making changes, post these changes as pull requests, and review your peers' pull requests. For more information on this process as well as how it'll be graded, please refer to the Code Review assignment located on Canvas.

Implementation

In this Sprint, you will be creating the new game sequence, including configuration and customization. There are four screens – welcome screen, initial configuration screen, game screen, and ending screen – you need to implement for Sprint 1. The following are the requirements:

1. Starting the application will open to a welcome screen for the application which:
    a. Has some way to start the game (e.g. a start button)
    b. Has some way to exit the game (not just by pressing the X in the upper right corner)
2. Starting the game should take the player to an initial configuration screen which:
    a. Requires input for the player to enter their name. The player should not be allowed to pass in an empty, null, or whitespaces-only name.
    b. Requires the player to choose their difficulty from at least 3 different options.
        i. Different difficulties should change the behaviors of the game in meaningful ways.

        ii. At a minimum, the selected difficulty should affect the player's starting health points (HP) and the amount of damage done to the player by enemies.
        iii. You will only be implementing the different starting HPs in this sprint. Damage done to the player by enemies will be implemented in a future sprint.
    c. Requires the player to choose their character sprite (image) from at least 3 different options.
    d. Allows the player to continue to the game screen after they input a valid name, choose a player model, and select a difficulty.
        i. The player must be able to visually see the name and the character they have chosen *before* they are allowed to move to the game screen.
3. The game screen should display graphically the first map the player will see. This is where most of the functionality in later sprints is implemented. For now, the screen must:
    a. Display player name and character sprite.
    b. Display chosen difficulty.
    c. Display player's starting health.
    d. Have a button to navigate to the ending screen. This button is temporary as it will be removed once game functionality is implemented in future sprints.
4. The ending screen should display the leaderboard and whether the player successfully completed the game or not. This will all be implemented in later sprints, so for now, the screen will be empty.
5. Make sure to abide by proper architecture practices (think about MVVM) while developing your project. While you will not be required to submit any proof of doing so in Sprint 1, we may ask for an explanation of your architecture in Sprint 2.

You have freedom over thematic elements of your game. For example, specific nouns (e.g., enemies, weapons) may be replaced by some other noun (e.g., animals, wands). Note that you are free to add functionality so long as you meet the requirements.

Testing Requirement

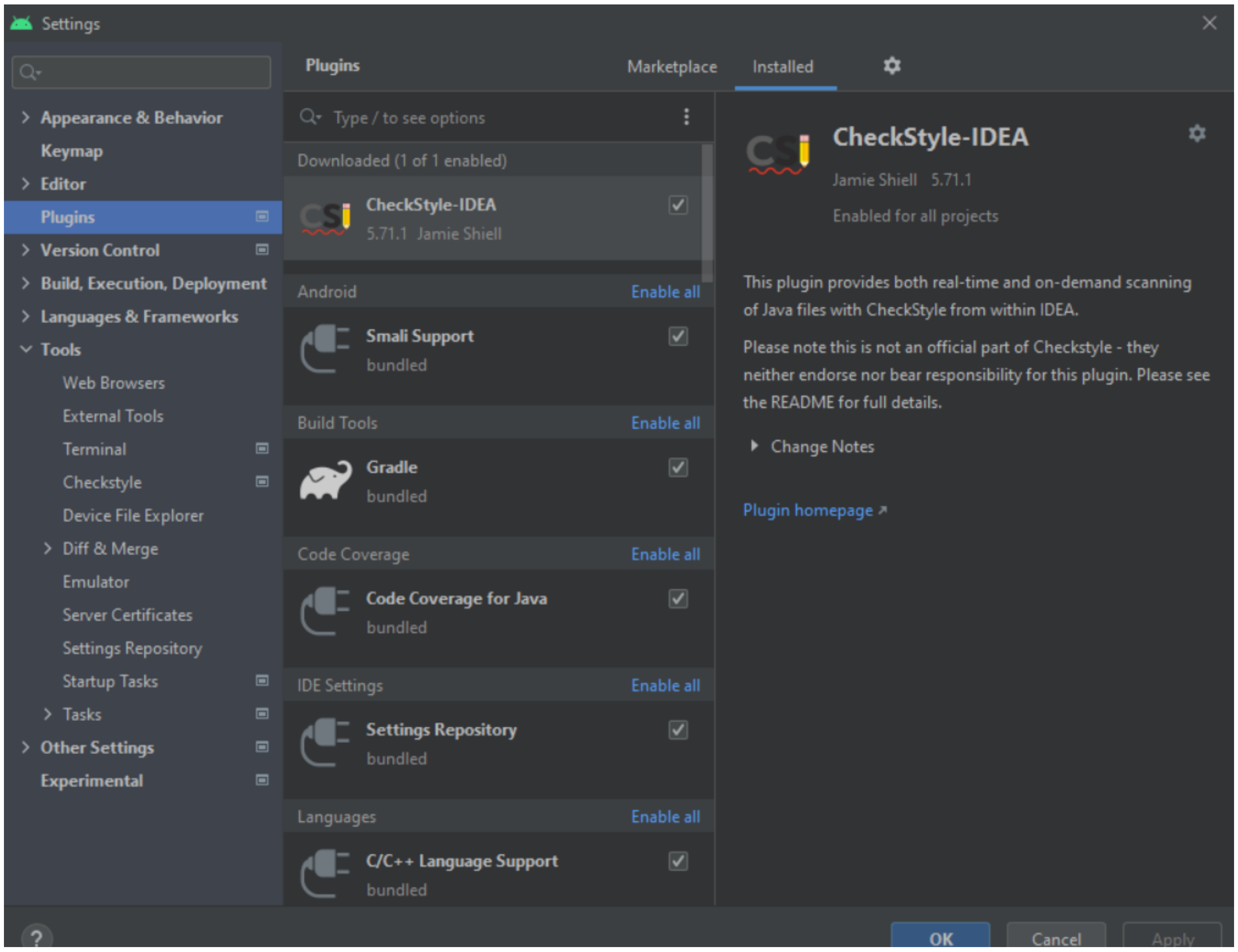
This sprint does not have a testing requirement, though subsequent sprints will. Create and develop your project with unit testing in mind. It may be wise to write some unit tests for the implementation requirements to practice for future sprints. Some example unit tests that one could prepare from Sprint 1 requirements are:

1. Detection of whitespace-only, null, and empty names
2. Player starting lives are different based on the chosen difficulty

Note: The UI of your team's game should be separate from the game's functionality/logic. These tests should not be composed of UI mocking, but instead, your team's tests should be validating your game's logic and event handling.

CheckStyle

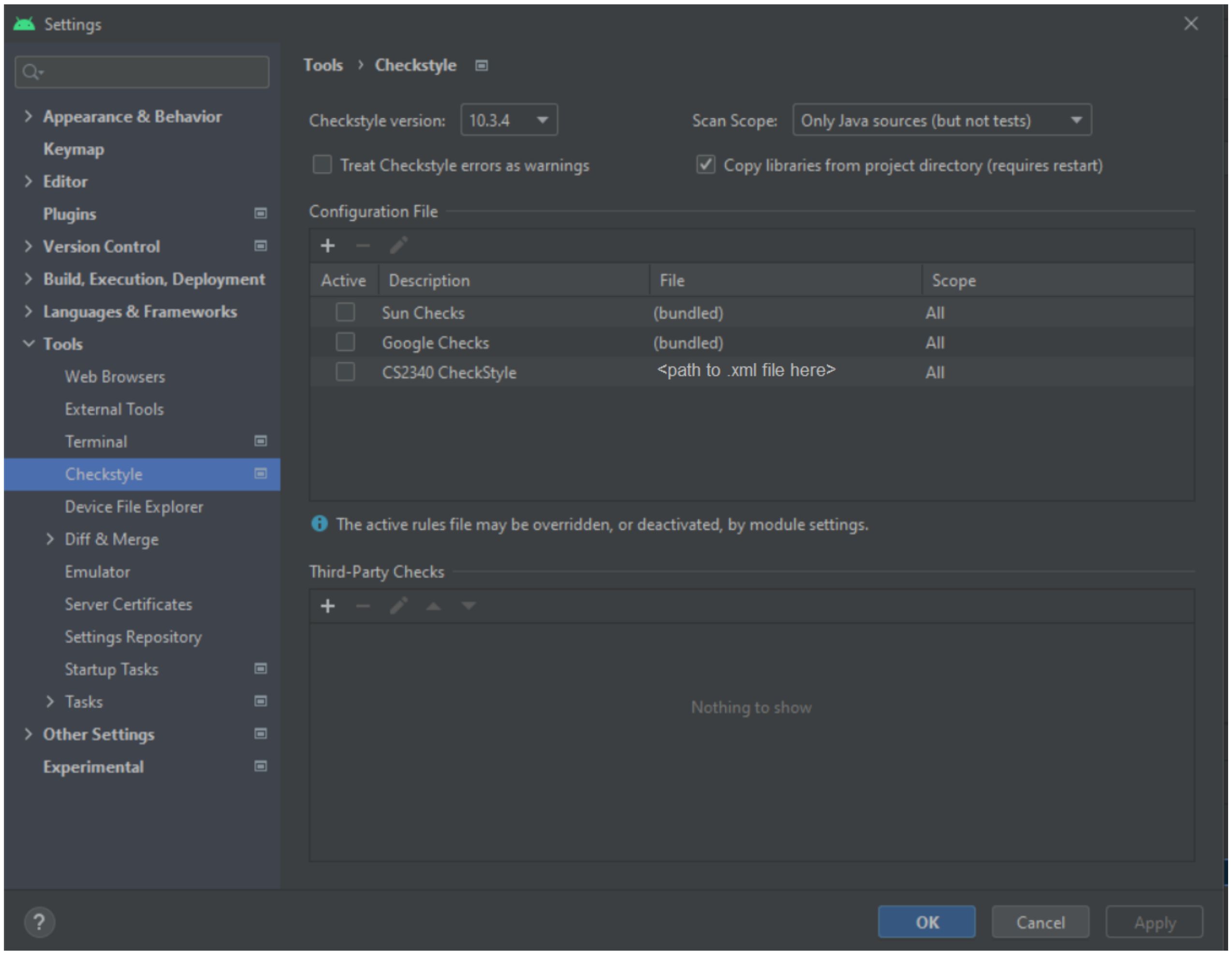
The Android Studio plugin CheckStyle-IDEA is required, as it enforces a consistent and readable code style. You can either download the plugin from this linked website and then install it from your disk in Android Studio, or you can install it directly from Android Studio. Both of these options can be done in the File > Settings > Plugins page in Android Studio. When prompted, restart Android Studio.



This plugin provides both real-time and on-demand scanning of Java files with CheckStyle from within IDEA.

Please note this is not an official part of Checkstyle - they neither endorse nor bear responsibility for this plugin. Please see the README for full details.

Now, navigate to File > Settings > Tools > Checkstyle. Set the CheckStyle version to 10.3.4 and add the CS2340 configuration file located here, which should be saved in a .xml file. This file is based off the one used in CS 1332, Data Structure and Algorithms, and is the coding standard for this class.


Settings

Tools › Checkstyle

Checkstyle version: 10.3.4

Scan Scope: Only Java sources (but not tests)

☐ Treat Checkstyle errors as warnings

☑ Copy libraries from project directory (requires restart)

Configuration File

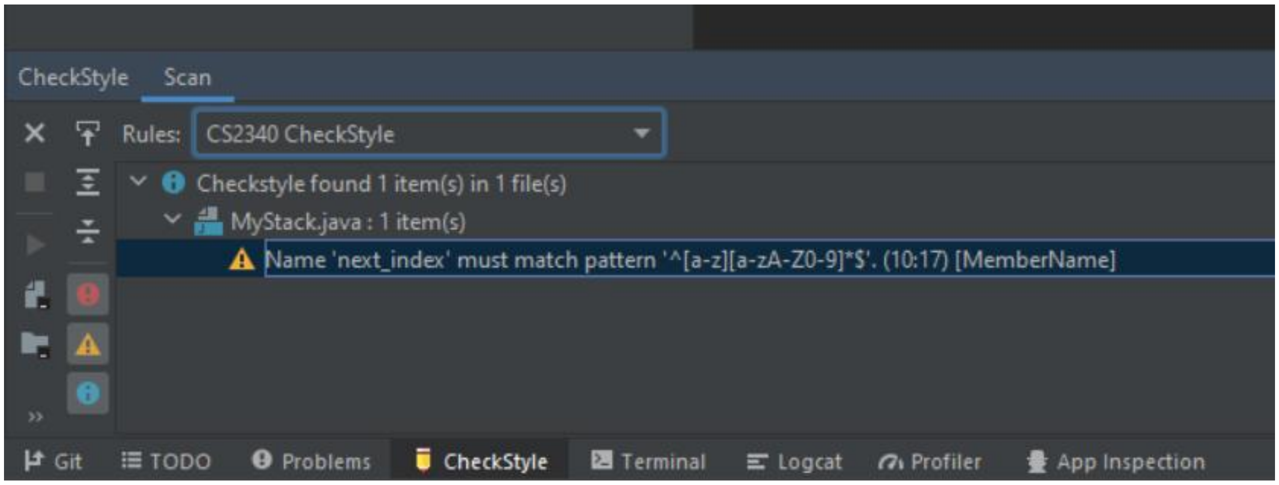
Active	Description	File	Scope
☐	Sun Checks	(bundled)	All
☐	Google Checks	(bundled)	All
☐	CS2340 CheckStyle	<path to .xml file here>	All

The active rules file may be overridden, or deactivated, by module settings.

Third-Party Checks

Nothing to show

You should now be able to run CheckStyle based on the CS2340 configuration through the CheckStyle tab at the very bottom left of Android Studio.



Rules: CS2340 CheckStyle

```
Checkstyle found 1 item(s) in 1 file(s)
  MyStack.java : 1 item(s)
    Name 'next_index' must match pattern '^[a-z][a-zA-Z0-9]*$'. (10:17) [MemberName]
```

Sprint Tagging

Tags are a way of marking a specific commit and are typically used to mark new versions of software. To do this, use "git tag" to list tags and "git tag –a tag name –m description" to create a tag on the current commit. **Important: To push tags, use "git push origin –tags"**. Pushing changes normally will not push tags to GitHub. You will be asked to checkout this tagged commit during demo. This is done with "git fetch --all --tags" and "git checkout tag name". **You will be required to pull this tag during the demo. If you forget to tag your commit or tag your commit late, you will be penalized with –5 points for each day late.**

Additional Resources

Under this section are some resources we think will help you and your team streamline the game creation process. If you have any questions about these resources, contact your TAs through Ed or office hours.

1. TA website: https://github.gatech.edu/pages/gtobdes/obdes/
2. Ed: https://edstem.org/us/courses/42802/discussion/
3. Android Studio Video (located on your section's canvas lecture page).

Demos

Your implementation of sprint features will be graded through an in-person TA demo. Demos will be held in CCB 267 the week after the sprint is due. Please sign up for a demo slot in Microsoft Bookings. Note that not all demo slots may be open at sprint release.

Summary

You will be graded according to the successful and correct completion of the design deliverables and implementation requirements above. Groups are required to demo to receive credit for the features they have implemented. This will be done during TA office hours after the due date of the design deliverables. TA demos will be able to be booked through Microsoft Bookings**.** You will submit your design deliverables as a combined PDF via the Canvas assignment. **You should also include a link to your GitHub project repository**. Your repository must be set to private. When you sign up for a demo, ensure that the shared TA GitHub account has been added to your repository **and you have tagged the code you intend to demo.** Points may be deducted if these guidelines are not followed.

Academic Integrity

Note on Plagiarism and the Use of AI Tools

We understand that technology and tools, including Artificial Intelligence platforms, have become readily accessible and can be valuable in various scenarios. However, students must be cautious about their usage in academic settings.

Please refer to the course policy regarding the use of AI tools for your projects. Using AI to generate or copy content for your project is not considered collaboration. Leveraging AI to produce work that you present as your own, without proper citation, is likely crossing the line into academic misconduct. It's essential to maintain integrity in all academic undertakings. If in doubt, always consult the course guidelines or reach out to the instructor on Ed Discussion for clarification.