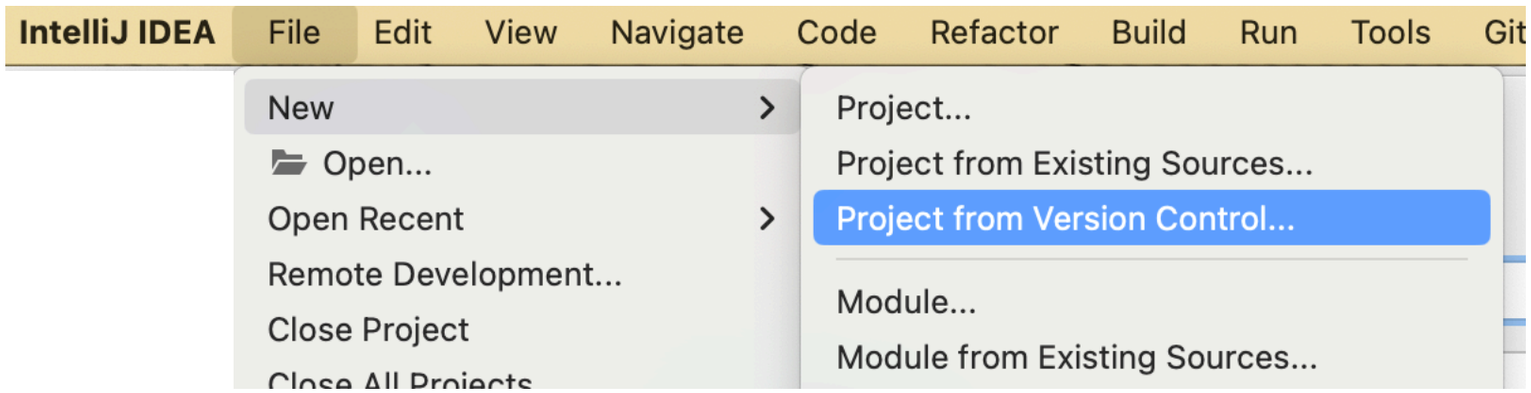
Klon et nyt projekt fra git repo



https://github.com/Dat1Cphbusiness/Game/

Obs!

IntelliJ vil default vælge IdeaProjects som directory Heri vil den default forsøge at oprette en ny mappe til dit nye projekt med samme navn som det projekt du er ved at klone (BankSystem). Du kan ændre det til noget andet, hvis du vil.

Developer A

2

Developer B

(Sværest)

Kig på klassediagram nr. 2. Her kan man se at der skal bygges en ny klasse, `FileIO`, med metoder til at gemme og indlæse data.

- Start med at oprette klassen `FileIO` og tilføj metodesignaturer for de to metoder i klassen, jvf. klassediagram nr. 2.

- Tilføj den nye metode `endSession()` i `Game`. Lad den være tom for nu.

- På sidste linje i `main` metoden lav et kald til Game klassens `endSession()`.

- I `Game` klassen, erklær og initialiser en instansvariabel `io`, af typen `FileIO`.

- Kald `saveData`, fra `endSession` metoden. Det skal se sådan ud:
`io.saveData(playerData, path, header);`

(I de næste trin laver I de tre parametre `(playerData, path, header)`, så udkommenter linjen for nu, så koden kompiler.)

I lavede det første parameter (`playerData`) i sidste trin, men I mangler `path` og `header` variablene:

`path` er stien til der hvor filen skal gemmes: `"data/playerData.csv"`. Så opret en folder(directory) med navnet "data", på samme niveau som src folderen (højreklik på roden af projektet)

`header` er det der skal stå på første linje i `playerData.csv` `"Name, Score"`

Nu næste trin skal I implementere - dvs. fylde kode i - `saveData` metoden.

Metoden `saveData` i `FileIO` kommer ikke til at kende noget til `Player` klassen. Den kan kun håndtere objekter af typen Strings. For at kunne gemme `Player` objekter, skal de derfor først laves om til `String` objekter.
I `Game` klassens `endSession`:

- Erklær en liste af strings som du kalder `playerData`.

- Gennemløb listen med `Player` objekter, `players`. Brug fx. et `for-each` loop: `for(Player p:players)`

- For hver iteration, lav en String med `p`'s 2 attributter, så den følger dette format: : `"Egon, 1000000"`. Genbrug evt. `Player` klassens `toString` metode.

- Føj stringen til `playerData` listen.

- Kig på den netop kaldte `toString()` metode og sikr dig at den returnerer det format vi er ude efter.

- lav en `try-catch` blok hvor `catch` delen fanger en `IOException`

- I `try` delen, lav en instans af `FileWriter` klassen, kald den `writer`. Sørg for at klassen bliver importeret. Med som argument skal den have `path` som du har i metodens parameterliste.

- Skriv headeren til filen:
```
writer.write(header+ "\n");
```

- Gennemløb `list` med et for-each loop og for hver iteration, skrives elementet til filen:
```
writer.write(s+ "\n");
```

- Efter loop'et, luk `writer` med `close`
- Test koden. Ingen data? Højreklik data folderen > Reload from Disk

Developer A

3

Vi skulle gerne have en tom metode i `FileIO` som hedder `readData`. Den implementerer vi nu. Metoden skal returnere et `ArrayList<String>` hvor hvert element indeholder en af linjerne fra `playerData.csv`

- Start med at initialisere den liste som skal returneres, kald den `data`.
- Lav også en instans af `File` klassen, som du kalder `file`. Med som argument skal den have `path`, (refererer til den sti, der bliver givet som argument, når metoden kaldes).
- Skriv en `try-catch` blok som håndterer en `FileNotFoundException`.
- I `try` delen, lav en instans af `Scanner` som skal læse fra filen:

```
Scanner scan  = new Scanner(file)
```

- Skriv signaturen for `startSession` metoden i `Game`.
- I toppen af metoden, initialiseres en liste, kald den `data` (ja, igen). Listen skal tildeles returværdien af et kald til `readData` metoden:

```
ArrayList<String> data = io.readData("data/
playerData.csv");
```

- Tjek om der er kommet elementer i `data`: `if(!data.isEmpty()){`
I så fald, skal vi gennemløbe listen med et for-each loop:
`for (String s:data) {…}`
Indholdet i loopet skrives i næste trin
- Hvis `data` derimod er tom, skal vi registrere spillere manuelt: Kald `registerPlayers()`
- I `main` metoden kan du nu erstatte kaldet til `registerPlayers` med et kald til `startSession()`

Developer B

- Stadig i `readData` metoden, skriv et `while` loop der kører sålænge dette statement er sandt: `scan.hasNextLine()`
- i kroppen af `while` loop'et skal linjerne en ad gangen, læses og gemmes i en variabel:

```
String line = scan.nextLine();
```

- Læg dernæst `line` i `data` (listen som I lavede i sidste trin).
- I `catch` delen kan printes en besked om at filen ikke blev fundet.

- For at headeren ikke også bliver taget med som kundedata, skal vi skippe den lige før `while` loopet kører:

```
scan.nextLine();//skip header
```

- returner til sidst listen, `data`

Fortsæt med at skrive disse linjer i kroppen af `for-each` loop'et

```
String[] values= s.split(",");
String name = values[0];
int score = Integer.parseInt(values[1].trim());
```

- Forklar hinanden hvad der sker i de tre linjer.
- Stadig inde i loop'et, brug de to variable `name` og `score` til at lave en ny instans af `Player`.
- Læg instansen i `players` listen.

- Test! Hvis det virker, kommer der ingen dialog mere, til gengæld printes samme data som der står i `playerData.csv`.