Battleships Online

Java Console Game

СЪДЪРЖАНИЕ

1. Архитектура

Конзолната игра Battleships online е имплементирана като клиент-сървър приложение. Бизнес логика на играта е имплементирана изцяло на сървърната част. Клиентът изпраща текстови съобщения до сървъра, които той се опитва да разчете като изпълними команди. Той пуска паралелно втора нишка, която слуша за съобщения, които сървъра може да му изпрати обратно. В този смисъл, клиентът е многонишков и представлява имплементация на обикновен chat-client.

1.1. Сървър

Сървърната имплементация се базира на Java NIO API-то, което позволява неблокиращи (асинхронни) входно-изходни операции. Това от една страна позволява ефективна работа на сървъра с огромен брой паралелни клиенти и заявки, а от друга страна, няма да е необходима синхронизация в сървъра, тъй като нишката, която ще изпълнява заявките ще е само една. Това, че само една нишка изпълнява командите на сървъра не е проблем, тъй като всяка команда се изпълнява или за константна времева сложност или за линейна по отношение на броя играчи, които са се свързали със сървъра (online) в момента на извикването ѝ.

1.2. Клиент

Клиентът, както вече споменахме по-горе е обикновен chat-client. При стартирането на клиент кода се очаква потребителя да въведе потребителското си име, с което да осъществи връзка със сървъра. (Забележете, че тук клиента само ще се осъществи връзка със сървъра със съответното потребителско име. Сървърът все още не пази състояние за този клиент). То се проверява чрез регулярен израз дали съдържа само малки или големи латински букви. Освен това, потребителското име трябва да е с размер не по-малък от 4 символа и не по-голям от 9 символа. В случай, че тези ограничения за потребителско име не са изпълнени, клиента дори не се опитва да се свърже със сървъра. Потребителя се приканва да въведе отново потребителско име, като му се съобщава, че предишното въведено от него потребителско име (username) не отговаря на изискванията.

При свързване със сървъра, клиента по подразбиране изпълнява коммандата set-username с подаден аргумент валиден (спрямо регулярния израз) username. Ако този username вече е зает и има друг потребител със същото потребителско име, тогава командата ще върне отговор, който ще индикира за това и клиента ще бъде приканен отново да въведе потребтелско име, с което да се опита да осъществи конекция със сървъра.

Чак след като сървърът върне съобщение, индикиращо, че клиента е успешно свързан с него, клиента пуска втора нишка, която работи паралелно с изпращащата команди нишка. Породената нишка, от своя страна, вместо да изпраща съобщения до сървъра – слуша дали сървъра иска да изпрати съобщение до нея и в момента, в който получи такова го принтира на конзолата на клиента.

2. Комуникация

Комуникацията между клиент и сървър се осъществява чрез следния протокол:
Клиента изпраща на сървъра команди под формата на обикновени стрингове. Сървърът има за грижа да провери дали тези стрингове отговарят на валидна команда, която може да се изпълни от него. В случай, че командата не е валидна, сървърът връща подходящо съобщение до клиента. Ако командата е валидна, след изпълнението ѝ, сървърът може да изпрати съобщение до повече от един клиент. И в двата случая, формата на съобщенията изпратени от сървъра са в JSON формат. На кратко – клиента изпраща на сървъра сурови стрингове, а сървъра му връща JSON стрингове.

3. Функционалности

3.1. man

User manual команда без аргументи. Форматира и показва online командите, които могат да се изпълнят на сървъра. Според това дали клиента е в игрови режим (host/guest) или само сърфира (online), се връща меню, което му показва какви са обичайните команди в режима, в който се намира. Например, ако е guest в някоя игра и изпълни команда man на сървъра – сървърът ще му върне ръководството на потребител, който е в режим на игра. Ако пък потребителят не е в режим на игра, тогава при изпълнение на командата man ще му се покажат какви са командите в не играл режим.

не игрови режим

```
Please enter your username: pesho
Trying to connect to Battle ships online with username pesho
Welcome to Battleships online!
> Your username is now pesho
battleships menu> man
> User manual in SURFING mode:
> man
> who-am-i
> set-username <username>
> change-username <username>
> list-users
> list-games
> register
> login <api-key>
> create-game <game-name>
> join-game {<game-name>}
> current-game
> save-game
> saved-games
> load-game <game-name>
> disconnect
```

игрови режим

```
battleships menu> create-game my-game
> You must login or register first
battleships menu> register
> You are now logged in with api-key: 821742744946511
battleships menu> create-game my-game
> Created game "my-game", players 1/2
battleships menu> man
> User manual in PLAYING mode:
> display
> place <[A-J]> <[1-10]> <[UP, RIGHT, LEFT, DOWN]> <[2-5]>
> place-all
> attack <[A-J]> <[1-10]>
> hacks
```

Потребител, който е в играл режим може да изпълнява и команди, които не са обичаини за играл режим, като например list-users, за да провери колко потребителя са свързани в момента със сървъра или list-games, за да провери кои са текущите създадени игри в момента на сървъра и т.н.

3.2. who-am-i

Команда без аргументи. Показва ефективното потребителско име, с което потребителя се е свързал към сървъра.

```
Please enter your username: 100yo
Accepts usernames only from 4 to 8 lowercase or uppercase latin letters!
Enter a valid username: stoyo
Trying to connect to Battle ships online with username stoyo
Welcome to Battleships online!
> Your username is now stoyo
battleships menu> who-am-i
> You are stoyo
```

3.3. set-username

Команда с един задължителен аргумент – <потребителско име>. Тази команда се изпълнява по подразбиране от всеки нов потребител, който се опитва да се свърже със сървъра. В случай, че подаденото като аргумент потребителско име вече е заето от друг потребител, сървърът очаква отново да се въведе ново свободно потребителско име.

```
Please enter your username: stoyo
Trying to connect to Battle ships online with username stoyo
Welcome to Battleships online!
> Username stoyo is taken, please select another username
Please enter your username: pesho
Trying to connect to Battle ships online with username pesho
> Your username is now pesho
battleships menu> set-username gosho
> Your username is already set to pesho
```

3.4. change-username

Команда с един задължителен аргумент – <ново потребителско име>. Тази команда може да се изпълнява само от клиент, който не е в режим на игра, а само е осъществил конекция със сървъра (online). При избор на ново потребителско име, което вече е заето, клиентът остава с текущото си потребителско име, с което първоначално се е свързал към сървъра и му се съобщва, че желаното ново потребителско име от него вече е заето. Ако клиента въведе отново текущото си потребителско име – получава съобщение, че вече е с това потребителско име.

```
Please enter your username: pesho
Trying to connect to Battle ships online with username pesho
Welcome to Battleships online!
> Your username is now pesho
battleships menu> change-username andy
> Username andy is taken, please select another username
battleships menu> change-username tosho
> Username changed. Your username is now tosho
battleships menu> who-am-i
> You are tosho
```

3.5. list-users

Команда без аргументи, която показва броя на всички свързани със сървъра потребители на първия ред, след което на всеки нов ред принтира текущото име на потребителя и неговия статус в сървъра (дали само сърфира или е в режим на игра).

```
> Total users connected now: 3
> desi is HOST
> pesho is ONLINE
> maria is GUEST
```

3.6. list-games

Команда без аргументи, която показва таблица с информация за наличните игри, които в момента се играят и обслужват от сървъра.

```
battleships menu> list-games
> | NAME                        | CREATOR     | STATUS      | PLAYERS     |
> |-----------------------------+-------------+-------------+-------------|
> | the battle of the vikings   | maria       | in progress | 2/2         |
> | my-game                     | pesho       | pending     | 1/2         |
```

3.7. register

Команда без аргументи, с която даден потребител се регистрира в сървъра, за да може да създава игра и да се включва като втори (guest) играч във вече създадени игри. Регистрацията се състои в това да се генерира уникален ключ от сървъра, който да се използва като парола от съответния потребител. Този ключ ще наричаме по-долу api-key. Той се образува от разликата, измерена в милисекунди, между текущото време и полунощ на 1 януари 1970 г. UTC (координирано универсално време) умножена по предварително зададено сучайно просто число (в случая използваме числото 499). Като подобрение, към ключа, който се връща при регистрация може да се добавят случайни символи в края му. Това ще подсили сигурността на ключа.

Наред с този уникален ключ, с който се идентифицира всеки регистриран потребител, се създава и запис в хеш таблица, който има за ключ уникалния api-key и стойност – нова хеш таблица с ключ – име на игра и стойност – клас SavedGame. Този клас SavedGame се използва само като структура от данни, която съхранява всичко необходимо за дадена игра, за да може да се възстанови отново, в случай че даден потребител иска да я довърши. Тоест всеки път когато даден потребител се регистрира – създаваме място, където да съхраняваме записаните му игри, които може да направи (в случай, че напусне сървъра, ако е в режим на игра заедно с друг потребител или просто иска да запамети игра, която играе, за да види как би се развила ако направи друг ход след като я зареди отново от там където я е запаметил).

```
battleships menu> register
> You are now logged in with api-key: 821763825190034
```

3.8. login

Команда с един задължителен аргумент <api-key>. С тази команда предварително регистриран потребител се свързва със сървъра. На сървъра се зареждат всички запаметени игри в оперативната памет заделена за този потребител. Тези игри се зареждат от файл във файловата система на сървъра, носещ същото име като съответния му api-key.

```
battleships menu> login 1234
> No such api-key
battleships menu> login 821763825190034
> You are now logged in with api-key: 821763825190034
```

3.9. create-game

Команда с един задължителен аргумент – <име на игра>. Тази команда е достъпна за изпълнение само за потребители, които са регистрирани с api-key в сървъра и не са в текуща игра в момента на изпълнението на командата. При опит за създаване на игра с име, което вече е заето от друга игра – се връща съобщение, което да индикира за това и не се създава нищо. Във всички останали случаи се създава нова игра с подаденото име, в която създателя е host, статуса на играта е pending и се чака втори играч да се присъедини. По време на изчакването може да се поставят корабчета (виж place командите по-долу)

```
Please enter your username: pesho
Trying to connect to Battle ships online with username pesho
Welcome to Battleships online!
> Your username is now pesho
battleships menu> create-game
> Requires one argument. Usage: create-game <game-name>
battleships menu> create-game "pesho's game"
> You must login or register first
battleships menu> login 821763825190034
> You are now logged in with api-key: 821763825190034
battleships menu> create-game "taken game-name"
> Game "taken game-name" already exists. Choose another game-name
battleships menu> create-game "pesho's game"
> Created game "pesho's game", players 1/2
```

3.10. join-game

Команда с един опционален параметър – <име на игра>. В случай, че този опционален параметър липсва, потребителя се присъединява към произволна pending игра. Ако аргумента е зададен, играча се опитва да се присъедини към играта с подаденото като аргумент име. Ако такава игра съществува и е със статус pending – присъединяването ще е успешно. В противен случай ще се индикира на потребителя чрез съобщение, че не е успял да се присъедини към такава игра. Аналогично на create-game командата и тази команда е достъпна само за потребители, които са се регистрирали с api-key в сървъра. Това е така, тъй като при успешно присъединяване към игра, тези потребители може да се disconnect-нат от сървъра по време на multiplayer (2/2) игра или да се опитат да запаметят текуща multiplayer (2/2) игра и трябва да има къде да се съхрани. Ако командата се извика от потребител, който вече е в игра, няма да се изпълни нищо друго освен да му се върне съобщение, което му напомня, че вече съществува текуща игра за този потребител.

```
battleships menu> join-game some-game
> You must login or register first
battleships menu> register
> You are now logged in with api-key: 821764375004701
battleships menu> join-game some-game
> There is no such game
battleships menu> join-game
> Joined game "pesho's game". Place your ships!
```

При успешно присъединяване към pending игра, на host играча се изпраща съобщение, с което да се индикира, че guest играч се е присъединил и играта може да започне.

```
battleships menu> > maria joined the game. Place your ships!
```

3.11. current-game

Команда без параметри. Връща текущата игра на потребител, който е регистриран и в момента е в дадена игра като създател или гост. Във всичко останали случай се връща съобщение от сървъра, индикиращо че потребителя не се намира в текуща игра.

```
battleships menu> current-game
> No current game
battleships menu> create-game "pesho's game"
> Created game "pesho's game", players 1/2
battleships menu> current-game
> You are currently HOST in pesho's game
```

3.12. save-game

Команда без параметри, която запаметява текущата игра на потребителя, който я извиква. Командата може да се извика, само ако текущата игра на потребителя е със статус in progress. Когато потребителя прекрати връзката си със сървъра (изпълни командата disconnect), тази команда (save-game) се изпълнява скрито (без да е извикана експлицитно от потребителя).

```
battleships menu> > maria joined the game. Place your ships!
battleships menu> save-game
> Game "pesho's game" saved
```

Ако потребителят вече има запаметена игра с името на текущата игра, играта не се запаметява и му се изпраща съобщение, което да му съобщи, че запаметяване на играта не се е състояло.

При запаметяване на игра, в структурата (класа) от данни SavedGame се съхраняват:

- името на играта;
- текущите състояния на двете дъски;

- бит, индикиращ за това дали host играче е наред;
- бит, индикиращ за това дали играча запаметил играта е бил host.

Това е минималната информация, която ни е необходима, за да може да възстановим запаметена игра.

3.13. saved-games

Команда без параметри, която показва текущите запаметени игри на регистриран потребител.

```
battleships menu> saved-games
> | NAME                        | CREATOR     | STATUS      | PLAYERS     |
> |-----------------------------+-------------+-------------+-------------|
> | pesho's game                | host        | saved       | 0/2         |
```

Забележете, че запаметените игри са с 0 играча, тъй като те са просто структури от данни, служещи за възобновяване на игра.

3.14. delete-game

Команда с един задължителен аргумент – <име на игра>, която да бъде изтрита. Командата може да се изпълни и с няколко аргумента – <игра 1> <игра 2> ... <игра n>. При извикването с няколко аргумента, сървърът ще се опита да изтрие последователно всяка игра с подаденото име, ако намери такава игра в запаметените от потребителя игри.

```
battleships menu> saved-games
> | NAME                        | CREATOR     | STATUS      | PLAYERS     |
> |-----------------------------+-------------+-------------+-------------|
> | pesho's game                | host        | saved       | 0/2         |
battleships menu> delete-game "some game" "pesho's game" other-game
> Trying to delete selected game(s)
> Could not delete "some game". No such game-name
> Game "pesho's game" was deleted
> Could not delete "other-game". No such game-name
battleships menu> saved-games
> | NAME                        | CREATOR     | STATUS      | PLAYERS     |
> |-----------------------------+-------------+-------------+-------------|
```

3.15. place

Команда с 4 задължителни параметъра – <[A-J]> <[1-10]> <[UP, RIGHT, DOWN, LEFT]> <[2-5]>. Тези параметри служат за позициониране на дадено корабче върху дъската.

- <[A-J]>: буква от A до J, указваща реда на дъската. Буквата може да е главна или малка (case-insensitive);
- <[1-10]>: число от 1 до 10, указващо клоната на дъската;
- <[UP, RIGHT, DOWN, LEFT]>: посока, която указва на къде ще се разполага корабчето. Посоката на разположение може да е с главни или малки букви или комбинация и от двете (case-insensitive);
- <[2-5]>: дължината на бойното корабче, която определя какъв ще е вида му. 2 – DESTROYER, 3 – CRUISER, 4 – BATTLESHIP, 5 – CARRIER.

Примерни извиквания:

- place B 2 RIGHT 2 – постава кораб с дължина 2, начална клетката B2 и посока – надясно
- place A 1 down 3 – поставя кораб с дължина 3, начална клетка A1 и посока – надолу
- place J 10 Up 4 – поставя кораб с дължина 4, начална клетка J10 и посока – нагоре
- place e 6 LeFt 5 – поставя кораб с дължина 5, начална клетка E6 и посока – наляво

Интерфейсът за поставяне на боен кораб е направен по този начин, за да е възможно най-гъвкав. Едно и също корабче може да се постави по 8 различни начина в общия слуай (ако не е ъглово и не е затиснато от други корабчета).

При извикването на тази команда се изпълняват проверки за:
- липсващ аргумент;
- статуса на извикалия я потребител (дали е host/guest в текуща игра);
- вече поставени всички корабчета;
- вече поставени всички корабчета от този тип;
- успешно конвертиране (парсване) на всички аргументи, като преди това се нормализират до малки букви, за да се постигне case-insensitive friendly интерфейс;
- позиционни аргументи (ред и колона), които излизат извън рамките на дъската;
- валидна дължина на корабче;
- излизане на корабчето извън рамките на дъската;
- поставяне на корабче върху вече съществуваща заст на друго корабче.

В случай, че всяка една от по-горе изброените проверки е преминала успешно – се поставя корабче на дъската.

Командите за поставяне на корабчета може да се изпълняват асинхронно и не зависят от това кой от двамата играчи е наред или дори дали има втори играч. При командите за атака, обаче, нещата стоят по друг начин (виж attack командите по-долу).

```
battleships menu> place b 2 right 2
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   - - - - - - - - - -
A |_|_|_|_|_|_|_|_|_|_|
B |_|X|X|_|_|_|_|_|_|_|
C |_|_|_|_|_|_|_|_|_|_|
D |_|_|_|_|_|_|_|_|_|_|
E |_|_|_|_|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|_|_|_|_|_|_|_|_|
J |_|_|_|_|_|_|_|_|_|_|
DESTROYER added
```

```
battleships menu> place e 4 up 4
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   - - - - - - - - - -
A |_|_|_|_|_|_|_|_|_|_|
B |_|X|X|_|_|_|_|_|_|_|
C |_|_|_|X|_|_|_|_|_|_|
D |_|_|_|X|_|_|_|_|_|_|
E |_|_|_|X|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|_|_|_|_|_|_|_|_|
J |_|_|_|_|_|_|_|_|_|_|
BATTLESHIP added
```

```
battleships menu> place e 6 left 3
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   - - - - - - - - - -
A |_|_|_|_|_|_|_|_|_|_|
B |_|X|X|X|_|_|_|_|_|_|
C |_|_|_|X|_|_|_|_|_|_|
D |_|_|_|X|_|_|_|_|_|_|
E |_|_|_|X|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|_|_|_|_|_|_|_|_|
J |_|_|_|_|_|_|_|_|_|_|
There already is a ship placed there
```

```
battleships menu> place i 10 left 5
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   - - - - - - - - - -
A |_|_|_|_|_|_|_|_|_|_|
B |_|X|X|X|_|_|_|_|_|_|
C |_|_|_|X|_|_|_|_|_|_|
D |_|_|_|X|_|_|_|_|_|_|
E |_|_|_|X|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|_|_|_|X|X|X|X|X|
J |_|_|_|_|_|_|_|_|_|_|
CARRIER added
```

```
battleships menu> place i 3 right 3
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   - - - - - - - - - -
A |_|_|_|_|_|_|_|_|_|_|
B |_|X|X|X|_|_|_|_|_|_|
C |_|_|_|X|_|_|_|_|_|_|
D |_|_|_|X|_|_|_|_|_|_|
E |_|_|_|X|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|X|X|X|X|X|X|X|X|
J |_|_|_|_|_|_|_|_|_|_|
CRUISER added
```

```
battleships menu> place j 10 down 2
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   - - - - - - - - - -
A |_|_|_|_|_|_|_|_|_|_|
B |_|X|X|X|_|_|_|_|_|_|
C |_|_|_|X|_|_|_|_|_|_|
D |_|_|_|X|_|_|_|_|_|_|
E |_|_|_|X|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|X|X|X|X|X|X|X|X|
J |_|_|_|_|_|_|_|_|_|_|
Out of border [DOWN]!
```

3.16. place-all

Команда без аргументи, която поставя на случаен принцип всички необходими корабчета на дъската на потребителя, който я е извикал. В случай, че потребителя вече е поставил всичките корабчета, се връща съобщение, което да му каже, че вече може да уведоми другия играч, ако такъв има и да започне процес на атакуване между двамата играчи.

Ако преди да извика тази команда, потребителя вече поставил няколко от наличните корабчета на дъската, командата ще донаслага останалите корабчета по случаен начин, като запази позициите на вече наличните на дъската корабчета.

Поставянето на корабчетата е абсолютно стохастично във всеки един сценарии, което прави изключително сложно да се пресметне времевата сложност на изпълнение на тази функция, но при арсенал от общо 30 клетки заети от корабчета от общо 100 клетки водна част – може да допуснем, че командата ще се изпълнява достатъчно бързо в средния случай.

```
battleships menu> place-all
> All ships were added, you are ready to attack!
GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10

   _ _ _ _ _ _ _ _ _ _
A |_|_|X|X|X|_|_|_|_|_|
B |_|_|_|_|_|_|X|X|_|_|
C |_|_|_|_|_|_|_|X|_|_|
D |X|X|X|X|_|_|X|_|_|_|
E |_|X|X|X|_|_|_|_|X|_|
F |_|X|X|X|_|_|_|_|X|_|
G |_|X|_|X|_|X|X|_|X|_|
H |_|X|_|X|_|X|_|_|X|_|
I |_|_|_|_|_|_|X|_|_|_|
J |_|_|_|_|_|_|X|_|_|_|
All ships are placed. Type "Start" to start the game
```

3.17. start

Команда без параметри, която служи за изпращане на съобщение до другия играч. Съобщението индикира за готовност на опонента да започне да атакува (вече е поставил всичките си корабчета на дъската). В случай, че друг играч няма, командата няма да се изпълни.

```
battleships menu> start
> Cannot start a single player mode
battleships menu> > maria joined the game. Place your ships!
battleships menu> start
> You are ready to start the battle
battleships menu> attack a 1
> One of the players did not press start
battleships menu> > maria is ready to start the battle
battleships menu> attack a 1
```

3.18. attack

Команда с два задължителни параметъра – <[A-J]> <[1-10]>, които са аналогични на първите два параметъра от командата place.

- <[A-J]>: буква от A до J, указваща реда на дъската. Буквата може да е главна или малка (case-insensitive);
- <[1-10]>: число от 1 до 10, указващо клоната на дъската;

Примерни извиквания:
- attack b 2 – атакува се клетката от ред B и колона 2 (клетката с координати B2)
- attack J 10 – атакува се клетката от ред J и колона 10 (най-долната и най-дясна клетка)

При извикване на командата се изпълняват проверки за:
- липсващ аргумент;
- статуса на извикалия я потребител (дали е в текуща игра);
- статуса на текущата игра (дали текущата игра е in progress);
- готовност на двата играча (дали и двамата играча в текущата игра са изпълнили команда start на сървъра успешно);
- реда на играча (дали е ред за атака на извикващия командата на сървъра);
- валидност на атаката и резултат от нея, който ще е необходим за отговор и на двата играча.

```
battleships menu> attack a 1
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |_|_|X|X|X|_|_|_|_|_|
B |_|_|_|_|_|X|X|_|_|_|
C |_|_|_|_|_|_|X|_|_|_|
D |X|X|X|X|_|_|X|_|_|_|
E |_|X|X|X|_|_|_|_|X|_|
F |_|X|X|X|_|_|_|_|X|_|
G |_|X|_|X|_|X|X|_|X|_|
H |_|X|_|X|_|X|_|_|X|_|
I |_|_|_|_|_|X|_|_|_|_|
J |_|_|_|_|_|X|_|_|_|_|
      ENEMY BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |-|_|_|_|_|_|_|_|_|_|
B |_|_|_|_|_|_|_|_|_|_|
C |_|_|_|_|_|_|_|_|_|_|
D |_|_|_|_|_|_|_|_|_|_|
E |_|_|_|_|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|_|_|_|_|_|_|_|_|
J |_|_|_|_|_|_|_|_|_|_|
You missed
```

```
battleships menu> attack d 5
> It's not your turn
battleships menu> > GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |_|_|X|*|X|_|_|_|_|_|
B |_|_|_|_|_|X|X|_|_|_|
C |_|_|_|_|_|_|X|_|_|_|
D |X|X|X|X|_|_|X|_|_|_|
E |_|X|X|X|_|_|_|_|X|_|
F |_|X|X|X|_|_|_|_|X|_|
G |_|X|_|X|_|X|X|_|X|_|
H |_|X|_|X|_|X|_|_|X|_|
I |_|_|_|_|_|X|_|_|_|_|
J |_|_|_|_|_|X|_|_|_|_|
      ENEMY BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |-|_|_|_|_|_|_|_|_|_|
B |_|_|_|_|_|_|_|_|_|_|
C |_|_|_|_|_|_|_|_|_|_|
D |_|_|_|_|_|_|_|_|_|_|
E |_|_|_|_|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|_|_|_|_|_|_|_|_|
J |_|_|_|_|_|_|_|_|_|_|
maria's last turn: A4
```

```
battleships menu> attack d 5
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |_|_|X|*|X|_|_|_|_|_|
B |_|_|_|_|_|X|X|_|_|_|
C |_|_|_|_|_|_|X|_|_|_|
D |X|X|X|X|_|_|X|_|_|_|
E |_|X|X|X|_|_|_|_|X|_|
F |_|X|X|X|_|_|_|_|X|_|
G |_|X|_|X|_|X|X|_|X|_|
H |_|X|_|X|_|X|_|_|X|_|
I |_|_|_|_|_|X|_|_|_|_|
J |_|_|_|_|_|X|_|_|_|_|
      ENEMY BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |-|_|_|_|_|_|_|_|_|_|
B |_|_|_|_|_|_|_|_|_|_|
C |_|_|_|_|_|_|_|_|_|_|
D |_|_|_|_|X|_|_|_|_|_|
E |_|_|_|_|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|_|_|_|_|_|_|_|_|
J |_|_|_|_|_|_|_|_|_|_|
You hit a CRUISER at position D5
```

За валидна атака се счита такава, която атакува в клетка в рамките на дъската, независимо дали на нея е имало корабче и ако е имало дали то е било вече атакувано на тази позиция или дори потопено изцяло. С други думи, за невалидна атака се счита само тази атака, която е с координати излизащи от границите на дъската или която има невалидни аргументи (брой/тип). При невалидна атака редът на играча не се сменя и той има възможност да атакува отново, докато не направи валидна атака.

3.19. hacks

Команда без параметри, която играе ролята на междинна команда или на команда-подсказка. При извикването и се показва начина, по който може да се използва скрита команда (attack-all <args> usage) или казано по-друг начин – команда, която не е типично да съществува за дадената игра, но е имплементирана с цел по-бързо тестване при демонстрация на играта. Тази команда hacks ще е валидна за извикване само от потребител, който е в режим на игра.

```
battleships menu> hacks
> attack-all <[A-J]> {<[A-J]>}...
```

3.20. attack-all

Команда с поне един задължителен аргумент – <[A-J]> и изброимо много опционални. Аргументите на тази команда указват буквите на редовете, които да бъдат изцяло атакувани. Тя не е легална команда, тъй като в оригиналната игра не би трябвало да съществува подобна команда, но тук съществува единствено с цел по-бърза презентация на играта. Чрез нея значително по-бързо може да се финализира дадена игра, като се симулират множество единични атаки. Зад тази команда стоят множество извиквания на командата attack с два аргумента, която разгледахме по-горе.

this player:

```
battleships menu> attack-all a b c d g j
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |-|_|_|_|_|_|X|X|_|_|
B |X|_|_|_|_|_|_|_|_|_|
C |X|_|X|X|_|_|_|_|X|_|
D |X|_|_|_|*|_|_|_|X|X|
E |X|_|_|_|X|_|_|_|X|X|
F |X|_|_|_|X|_|X|_|X|X|
G |_|_|_|_|_|X|X|_|_|_|
H |_|_|_|_|_|X|X|_|_|_|
I |_|_|_|X|X|X|X|_|_|_|
J |_|_|_|_|_|_|_|X|X|_|
      ENEMY BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |-|-|X|X|X|-|-|-|-|-|
B |-|-|-|-|-|X|X|-|-|-|
C |-|-|-|-|-|-|X|-|-|-|
D |X|X|X|X|-|-|X|-|-|-|
E |_|_|_|_|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |-|X|-|X|-|X|X|-|X|-|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|_|_|_|_|_|_|_|_|
J |-|-|-|-|-|X|-|-|-|-|
You hit 16 time(s) and destroyed 5 ship(s)
```

other player:

```
battleships menu> > GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |-|-|*|*|*|-|-|-|-|-|
B |-|-|-|-|-|*|*|-|-|-|
C |-|-|-|-|-|-|*|-|-|-|
D |*|*|*|*|-|-|*|-|-|-|
E |_|X|X|X|_|_|_|_|X|_|
F |_|X|X|X|_|_|_|_|X|_|
G |-|*|-|*|-|*|*|-|*|-|
H |_|X|_|X|_|X|_|_|X|_|
I |_|_|_|_|_|_|X|_|_|_|
J |-|-|-|-|-|*|-|-|-|-|
      ENEMY BOARD
   1 2 3 4 5 6 7 8 9 10
   _ _ _ _ _ _ _ _ _ _
A |-|_|_|_|_|_|_|_|_|_|
B |_|_|_|_|_|_|_|_|_|_|
C |_|_|_|_|_|_|_|_|_|_|
D |_|_|_|_|X|_|_|_|_|_|
E |_|_|_|_|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|_|
H |_|_|_|_|_|_|_|_|_|_|
I |_|_|_|_|_|_|_|_|_|_|
J |_|_|_|_|_|_|_|_|_|_|
You were massively attacked and hit 16 time(s) and lost 5 ship(s)!
```

3.21. display

Команда без аргументи, която показва състоянието на текущата игра. В случай, че играта е pending и още няма противник (guest), се презентира само дъската на извикалия командата. Ако играта е in progress и има втори игра, се презентират двете дъски, като тази на противника е кодирана, т.е. клетките с усшешни атаки на противникови кораби са отбелязани с "X", а клетките с липсващи кораби, които са атакувани (неуспешните атаки) са отбелязани с тире "–".

```
battleships menu> display
> GAME STATE:
       YOUR BOARD
   1 2 3 4 5 6 7 8 9 10

   – – – – – – – – – –
A |-|-|*|*|*|-|-|-|-|-|
B |-|-|-|-|-|*|*|-|-|-|
C |-|-|-|-|-|-|*|-|-|-|
D |*|*|*|*|-|-|*|-|-|-|
E |_|X|X|X|_|_|_|_|X|_|
F |_|X|X|X|_|_|_|_|*|-|
G |-|*|-|*|-|*|*|-|*|-|
H |_|X|_|X|_|X|_|_|X|_|
I |_|_|_|_|_|X|_|_|_|_|
J |-|-|-|-|-|*|-|-|-|-|
      ENEMY BOARD
   1 2 3 4 5 6 7 8 9 10

   – – – – – – – – – –
A |-|-|-|-|-|-|X|X|-|-|
B |_|_|_|_|_|_|_|_|_|_|
C |X|-|X|X|-|-|-|-|X|-|
D |_|_|_|_|X|_|_|_|_|_|
E |_|_|_|_|_|_|_|_|_|_|
F |_|_|_|_|_|_|_|_|_|_|
G |_|_|_|_|_|_|_|_|_|-|
H |-|-|-|-|-|X|X|-|-|-|
I |-|-|-|X|X|X|X|-|-|-|
J |-|-|-|-|-|-|-|X|X|-|
```

3.22. disconnect

Команда без параметри. Въпреки отсъствието на параметри и това, че я разглеждаме последна, тази команда е една от най-комплексните и сложни от към бизнес логика команди, които се изпълняват на сървъра. При извикването на тази команда от клиента, независимо каква бизнес логика ще се изпълни на сървъра и какво съобщение ще се върне от сървъра, клиента, който е извикал командата ще изпечати съобщението на конзолата си и ще терминира програмата си (ще спре изпълнението на двете си клиентски нишки).

Бизнес логиката на тази команда на сървъра е по-интересна. Първото нещо което се проверява е дали потребителя, който напуска сървъра е бил в режим на игра, която е със статус in progress, т.е. дали играте е била multiplayer (2/2) и е имал и противник. Ако това не е изпълнено и играта не е била multiplayer (2/2), се премахва remote socket address-а на socket-а от socket channel-а, с който потребителя се е свързал към сървъра и играта не се записва.

Но, ако потребителя е бил в текуща игра, която е била in progress и е изпълнил командата disconnect, то тогава играта се трансформира от in progress на pending, като се актуализира статута на противниковия играч (ако е бил guest става на host), акуализират се socket channel-ите и дъските на играчите, както и правилния ред и чак тогава се записва играта в оперативната памет със запаметените игри. По този начин, след записа, ще сме извлекли цялата необходима информация от играта, за да може да я възстановим коректно при повторно влизане на потребителя с неговия api-key.

След това се записват всички запаметени игри от хиип паметта за клиента в текстов файл на сървъра, който има същото име както съответния му api-key ключ.

Хиипа се освобождава, за да не заема място на сървъра и за да може заявките отнемащи линейна времева сложност да се изпълняват по-бързо.

При напускане на един от играчите, на срещуположния играч се показва съобщение, което да му индикира, че той е останал в играта на single player mode.

this player:

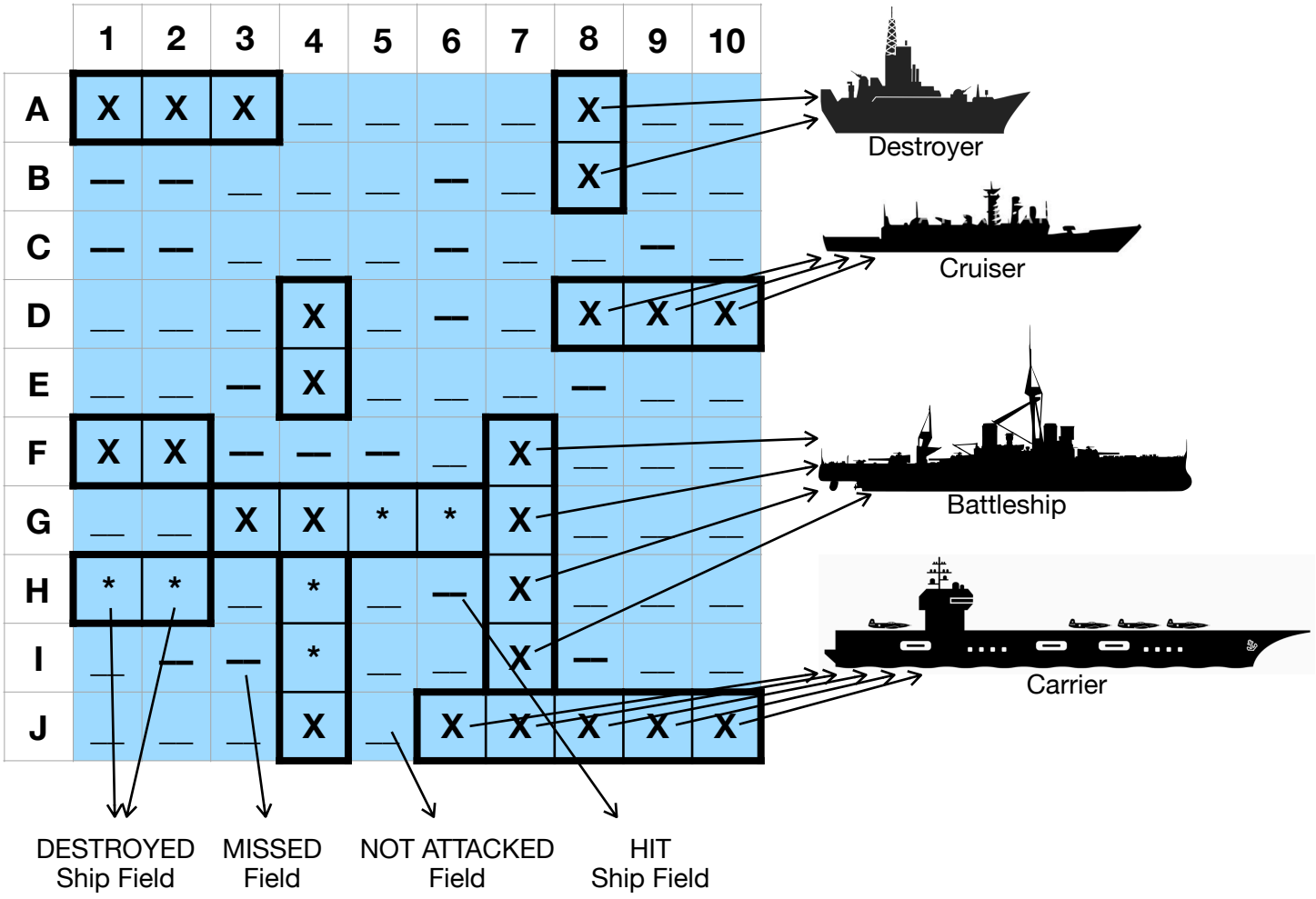
```
battleships menu> disconnect
See you soon on the battlefield pesho
Disconnected from server

Process finished with exit code 0
```

other player:

```
battleships menu> > pesho exit the game. Waiting for another player to join...
battleships menu>
```

4. Представяне на данните

4.1. Дъска (Board)



Дъската от играта Battleships Online е представена като структура от данни (клас) Board с множество вложени зависимости от други класове. Дъската представлява квадратна матрица от класове Field, които имат член данни изброим тим FieldType (enum) и клас Ship. Освен тази матрица, дъската има и член данна – хеш таблица с ключ число и стойност списък от корабчета (Ship) и totalDestroyedShips от целочислен тип. Числото от ключа на хеш таблицата представлява дължината на корабчето, а стойността е клас, който представлява корабче.

Класът Ship съдържа член данни ShipType от изброим тип, shipDeck от целочислен тип и списък от променливи от тип Coordinate. Coordinate променливите имат две член данни от целочислен тип, указващи координатите на които е разположен този кораб. shipDeck е брояч, който показва колко от клетките на кораба са останали здрави (не атакувани). Очевидно когато този брояч стане равен на 0 – корабчето ще е потопено изцяло.

* Както се вижда от фигурата по-горе, когато имаме кораб разположен на дъската, няколко съседни клетки сочат към един и същ клас Ship, който представлява кораб. В този смисъл, списъка от координати, на които е разположен съответния кораб, е съществено важна член данна на класа Ship, тъй като чрез нея ще може да възстановим коректно дъската от записана като JSON стринг дъска. Във функцията adjustBoard от класа LoadGame се случва именно това – за всеки един от корабите на дъската, за всяка една от неговите координати, се пренасочва да сочи към референцията на съответния клас Ship.

Наличието на множество вложени зависимости в Board налагат и използването на mock-ове, с които се справяме със зависимостите при unit тестването.

4.2. Потребител (Player)

Играчът е представен от клас Player, който има член данни username от тип String за потребителското му име, id от тип String за идентификационния номер на сървъра, който представлява socketAddress-а, с който се е свързал, дъска (клас Board), която разгледахме по-горе в 4.1. и SocketChannel, който ни е необходим, за да знае сървъра на кой канал да изпрати съобщение при необходимост.

4.3. Игра (Game)

Играта е представена от клас Game, който има член данни два играча от тип Player, които разгледахме по-горе в 4.2., име на играта name от тип String, две променливи hostReady и guestReady от тип boolean, указващи дали съответните играчи са наслагали всичките си корабчета и са готови за атака, статус на играта status от изброим тип GameStatus, булева поменлива hostTurn, която указва дали host играча е на ред и дъска savedBoardSecondPlayer, която сочи към дъска в някакво състояние, ако играта е заредена от запаметена игра или сочи към null, ако играта е новосъздадена. Когато даден потребител се опита да се присъедини към дадена игра, първото нещо което прави метода join от Game е да провери дали savedBoardSecondPlayer не сочи към null, ако не сочи, присвоява тази дъска към която сочи на присъединяващия се играч, ако пък сочи към null му създава нова дъска (т.е. играта не е заредена от запаметена игра).

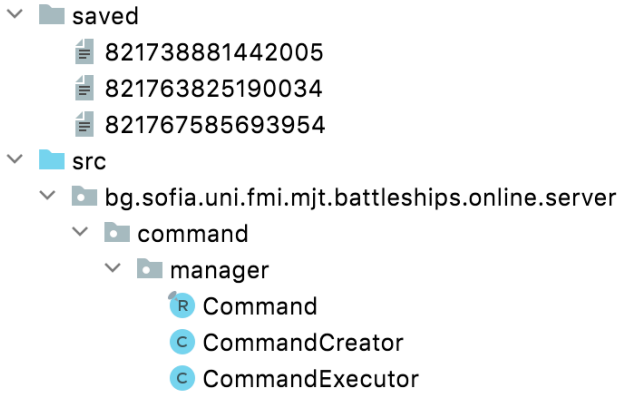
4.4. Сървърно хранилище за данни (Storage)

Хранилището за данни е клас, в който са обединени всички данни, които са необходими на сървъра, за да може да осъществява бизнес логиката на играта Battleships Online. Това са две хеш таблици socketAddressUserIn и apiKeySavedGames.

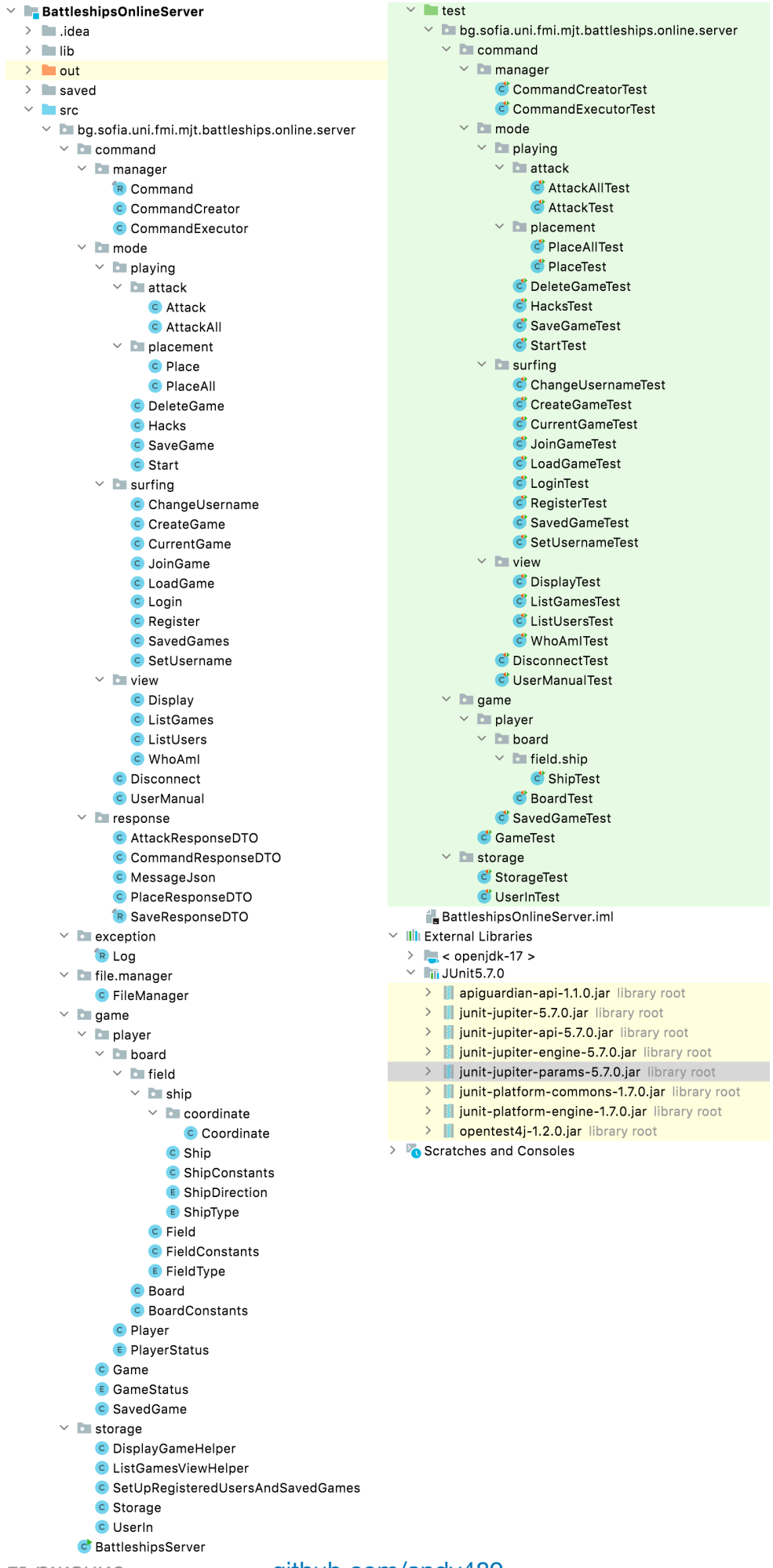
socketAddressUserIn: използва за ключ String, който представлява socketAddress-а на всеки осъществил конекция със сървъра (ползваме го за id), а за стойност използва UserIn клас, който представлява всичката необходима информация за даден потребител, който в момента има връзка със сървъра. UserIn класа има за член данни: потребителско име username от тип String, потребителски статус (status) от изброим тип PlayerStatus, apiKey ключ от тип String и клас Game, който разгледахме по-горе в 4.3.

apiKeySavedGames: използва за ключ String, който представлява apiKey ключ на регистриран в сървъра потребител, а за стойност използва друга хеш таблица с ключ име на игра от тип String и стойност SavedGame структура, която разгледахме по-горе.

ВАЖНО! В конструктура на класа BattleshipsServer отговарящ за сървърната част се инициализира CommandExecutor клас с празен Storage клас. Този CommandExecutor се грижи за изпълнението на командите на сървъра и в инициализацията му се извиква статитчен метод на класа SetUpRegisteredUsersAndSavedGames, който прочита от файловата система на сървъра всички файлове от папка saved, които имат за имена apiKey ключовете на регистрирали се на сървъра потребители и в тях има хеш таблици от запаметени игри в JSON формат. С тази информация сървъра актуализира хранилището си за данни при стартиране, за да няма загуба на данни в случай, че се наложи да го рестартираме.



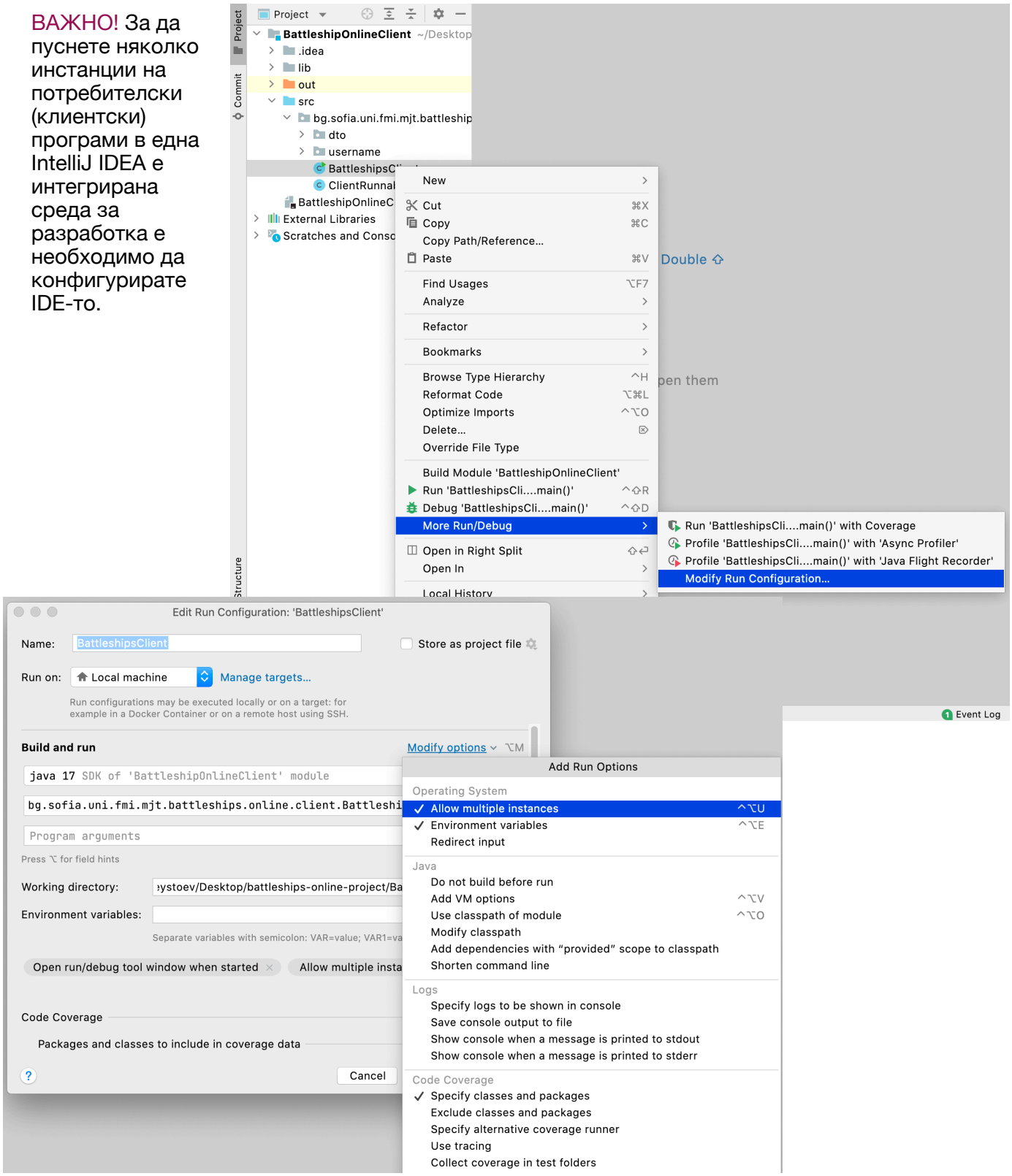
5. Структура

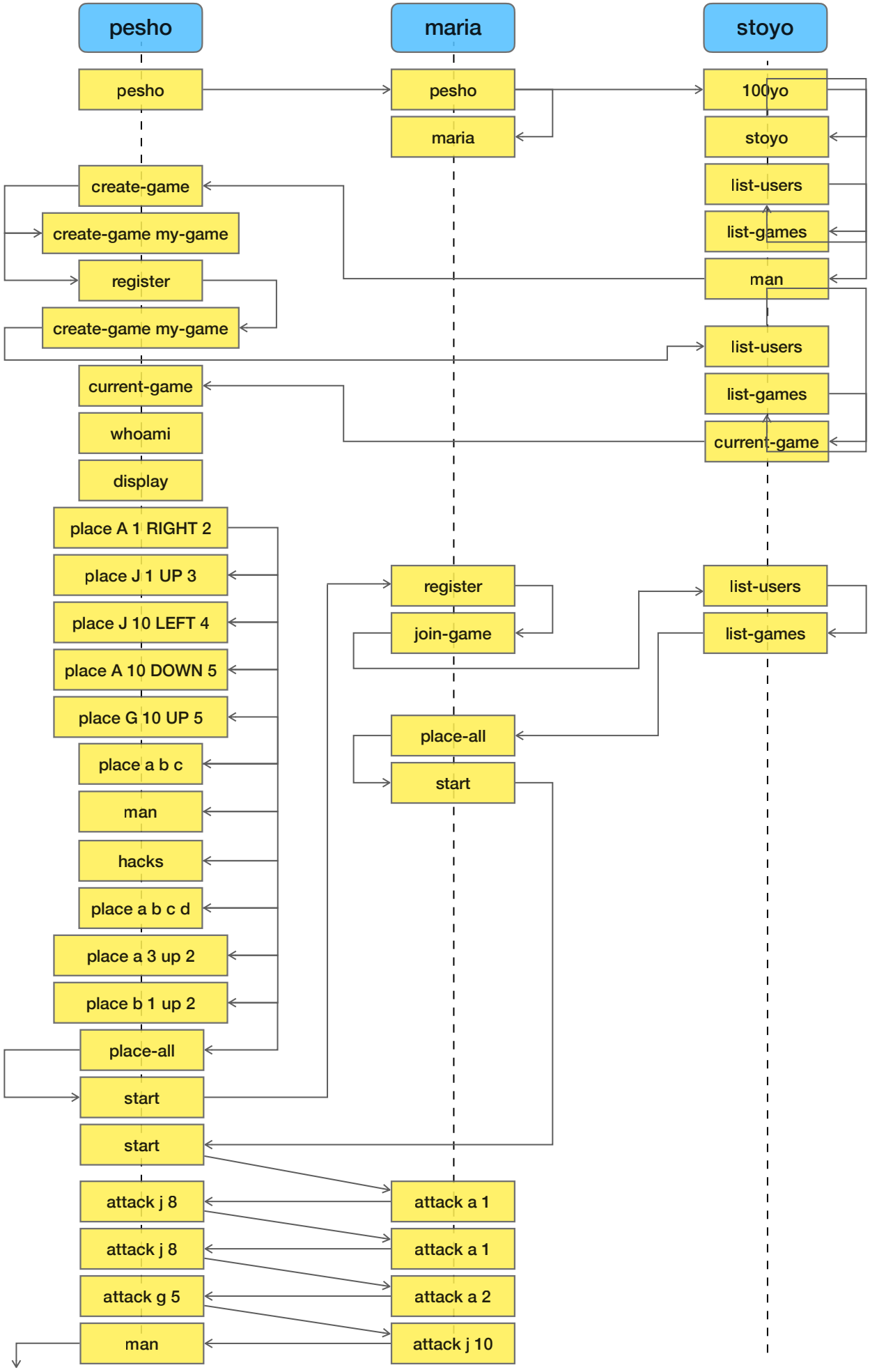

BattleshipsOnlineServer
.idea
lib
out
saved
src
bg.sofia.uni.fmi.mjt.battleships.online.server
command
manager
Command
CommandCreator
CommandExecutor
mode
playing
attack
Attack
AttackAll
placement
Place
PlaceAll
DeleteGame
Hacks
SaveGame
Start
surfing
ChangeUsername
CreateGame
CurrentGame
JoinGame
LoadGame
Login
Register
SavedGames
SetUsername
view
Display
ListGames
ListUsers
WhoAmI
Disconnect
UserManual
response
AttackResponseDTO
CommandResponseDTO
MessageJson
PlaceResponseDTO
SaveResponseDTO
exception
Log
file.manager
FileManager
game
player
board
field
ship
coordinate
Coordinate
Ship
ShipConstants
ShipDirection
ShipType
Field
FieldConstants
FieldType
Board
BoardConstants
Player
PlayerStatus
Game
GameStatus
SavedGame
storage
DisplayGameHelper
ListGamesViewHelper
SetUpRegisteredUsersAndSavedGames
Storage
UserIn
BattleshipsServer
test
bg.sofia.uni.fmi.mjt.battleships.online.server
command
manager
CommandCreatorTest
CommandExecutorTest
mode
playing
attack
AttackAllTest
AttackTest
placement
PlaceAllTest
PlaceTest
DeleteGameTest
HacksTest
SaveGameTest
StartTest
surfing
ChangeUsernameTest
CreateGameTest
CurrentGameTest
JoinGameTest
LoadGameTest
LoginTest
RegisterTest
SavedGameTest
SetUsernameTest
view
DisplayTest
ListGamesTest
ListUsersTest
WhoAmITest
DisconnectTest
UserManualTest
game
player
board
field.ship
ShipTest
BoardTest
SavedGameTest
GameTest
storage
StorageTest
UserInTest
BattleshipsOnlineServer.iml
External Libraries
< openjdk-17 >
JUnit5.7.0
apiguardian-api-1.1.0.jar library root
junit-jupiter-5.7.0.jar library root
junit-jupiter-api-5.7.0.jar library root
junit-jupiter-engine-5.7.0.jar library root
junit-jupiter-params-5.7.0.jar library root
junit-platform-commons-1.7.0.jar library root
junit-platform-engine-1.7.0.jar library root
opentest4j-1.2.0.jar library root
Scratches and Consoles

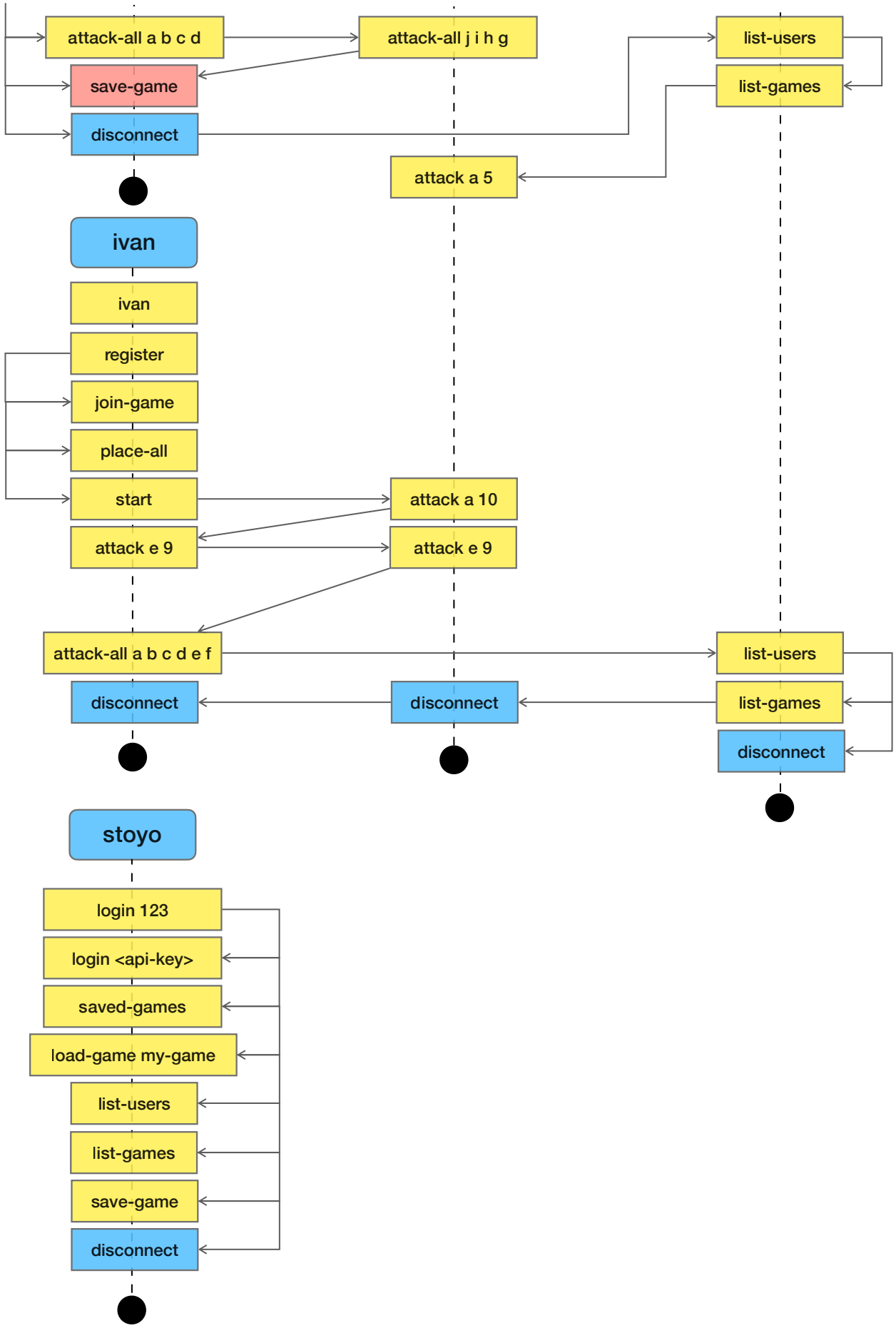
6. Примерен тестови сценарии

Тестовият сценарии представлява една псевдо диаграма на последователностите, която има за цел единствено да ръководи презентирането на проекта. Диаграмата заимства някой нотации от sequence диаграмите, но в никакъв случай не може да бъде разглеждана като такава. Това е така, тъй като в нашия случай различните потребители са паралелни програми, но за улеснение са показани като фон нойманова последователност. Тази последователност е репрезентация на последователността, която искаме да симулираме. Целта ѝ е да обхване някой клюови казуси и проблеми, които са разрешени чрез архитектурни, структурни и дизайн решения преди и по време на имплементацията на играта.

ВАЖНО! За да пуснете няколко инстанции на потребителски (клиентски) програми в една IntelliJ IDEA е интегрирана среда за разработка е необходимо да конфигурирате IDE-то.




pesho
maria
stoyo
pesho
pesho
maria
100yo
stoyo
list-users
list-games
man
list-users
list-games
current-game
create-game
create-game my-game
register
create-game my-game
current-game
whoami
display
place A 1 RIGHT 2
place J 1 UP 3
place J 10 LEFT 4
place A 10 DOWN 5
place G 10 UP 5
place a b c
man
hacks
place a b c d
place a 3 up 2
place b 1 up 2
place-all
start
register
join-game
list-users
list-games
place-all
start
start
attack j 8
attack j 8
attack g 5
man
attack a 1
attack a 1
attack a 2
attack j 10


attack-all a b c d
attack-all j i h g
list-users
save-game
list-games
disconnect
attack a 5
ivan
ivan
register
join-game
place-all
start
attack a 10
attack e 9
attack e 9
attack-all a b c d e f
list-users
disconnect
disconnect
list-games
disconnect
stoyo
login 123
login <api-key>
saved-games
load-game my-game
list-users
list-games
save-game
disconnect