# Get15 Project

# Имплементация

Проектът включва интерфейс Get15Interface, който декларира методи:

* connect за свързване на клиента(играча) към RMI сървъра и определяне на цвета, с който ще играе;
* takeNumber, с който клиентът информира сървъра за числото, което избира и с какъв цвят играе, тоест с какъв цвят маркира избраното число;
* disconnect, с който клиентът информира сървъра за изключването си от играта при затваряне на прозореца.

Имплементацията на този интерфейс се съдържа в Get15Implementation и дефинира логиката на играта. Той съдържа референции redPlayer и bluePlayer, които се използват за callback на играчите и дефинират цветовете, с които играят играчите. Също така Get15Implementation съдържа масив от символи с 9 елемента – по 1 за всяко число в играта от 1 до 9. Тоест елементът с индекс x съответства на числото x+1. Всеки от тези символи показва дали числото не е избрано от нито един играч (символ ‘ ‘), числото е избрано от играча с червен цвят (символ ‘r’) или числото е избрано от играча със син цвят (символ ‘b’).

Get15Implementation има два конструктора – по подразбиране и дефиниращ порт, на който се експортира обекта. И двата конструктора извикват метода initializeNumbers(), който служи за инициализиране на масива със символи с ‘ ‘, защото в началото никое число не е избрано.

Методът connect съдържа дефиниция на наследения от Get15Interface метод. Той инициализира redPlayer или bluePlayer с подадения аргумент в зависимост кой от тях още не е инициализиран. Ако и двата са инициализирани, то клиентът се уведомява със съобщение, че вече има двама играчи в играта и той няма как да се включи.

Методът takeNumber съдържа дефиниция на наследения от Get15Interface метод. В него първо се прави проверка дали числото вече е избрано, в който случай се връща false. В противен случай числото, зададено като първи аргумент, се маркира като взето от играча с цвят, зададен с втория аргумент, като се промени стойността на съответния елемент в масива numbers. След това ако някой от играчите е затворил своя графичен прозорец, се връща false. След това се правят проверки дали някой от играчите не е спечелил с private метод isWon и дали не са избрани всички числа без да има победител с private метод allNumbersAreTaken, като в тези случаи играта приключва и се изпращат съобщения до клиентите за състоянието.

В main метода се регистрира отдалечения обект под името Get15Implentation и се извежда съобщение, че е регистриран на конзолата.

Интерфейсът CallBack декларира методи:

* takeTurn, с който сървъра информира клиента дали е негов ред;
* notify, с който сървъра изпраща съобщение до клиента, което да бъде показано в неговия графичен прозорец;
* mark, с който сървъра информира клиента за избора на число на другия играч.

CallBackImplementation имплементира този интерфейс и има референция thisClient от тип Get15ClientRMI, с която клиентът се извиква от сървъра чрез callback механизъм. В този клас се дефинират всички наследени методи да извикват съответния метод на thisClient, който изпълнява съответното действие.

Get15ClientRMI наследява Application и дефинира графичното приложение на клиента. Съдържа цвета на играча (color), това дали е негов ред (myTurn), референция към сървъра (get15) и елементите на графичния прозорец:

* етикет, който чрез низ показва цвета на играча по време на играта (lblColor);
* етикет, който показва последното изпратено от сървъра съобщение – кой играч е на ред, дали си спечелил или загубил играта и дали всички числа вече са избрани (lblMessage);
* девет бутона – по един за всяко число, което може да бъде избрано (numbers – масив от девет елемента от тип Button).

Методът initializeRMI служи за намиране на отдалечения RMI обект в регистъра и съответно инициализирането на член-данната get15 и за свързване на клиента към този сървър чрез метода connect, тоест сървъра определя и цвета на играча и той се запазва в член-данната color и за извеждане на съобщение за цвета на играча в етикета lblColor.

Методите setMyTurn, setMessage и mark служат съответно за определяне дали реда е на този играч, за извеждане на получено от сървъра съобщение в lblMessage, за маркиране на бутона със зададеното в първия аргумент число със зададения във втория аргумент цвят чрез оцветяването му. И трите се извикват от сървъра чрез callback механизъм.

В start метода се определят разположението и размерите на графичния прозорец и елементите му, както и формата на бутоните – за по-естетично се използват кръгли бутони. Задава се събитие, което да се случва при натискането на всеки от бутоните:

* ако числото вече е избрано, нищо не се случва;
* ако редът е на другия играч, нищо не се случва;
* ако редът е на този играч и числото не е избрано, бутонът с избраното число се оцветява в цвета на играча.

Извиква се и методът initializeRMI и се задава клиентът да извиква методът disconnect на сървъра при затваряне на прозореца.

# Тестване на кода

Тъй като е важно графичният интерфейс на една игра да е user-friendly и интуитивен, то кодът беше тестван от хора, които не знаят какво точно трябва да правят освен че искат да получат сума 15 на кои да е 3 избрани от тях числа, за да се види дали дизайнът, съобщенията и цветовете са достатъчно ориентировъчни за незапознатия без да се четат допълнителни инструкции.

Също така проектът е тестван чрез симулиране на различни сценарии на играта:

* опит за избиране на число, което вече е избрано;
* опит за избиране на число, когато не е ред на този играч;
* симулиране на игра, в която единия побеждава, за да се види дали се изписват правилните съобщения и дали се блокират и двамата играчи;
* симулиране на игра без победител, в която всички числа са избрани, за да се види дали се изписва правилното съобщение и дали се блокират играчите;
* затваряне на прозореца на единия играч преди играта да е свършила, за да се види дали се отчита, че той е извън играта.

# Ограничения на кода

Поради правилата на играта, кодът е ограничен до създаване на конкретните 9 бутона за числата от 1 до 9 и до двама играчи, които могат да играят във всеки един момент, тоест в даден момент може да се играе само една игра.

# Решени и нерешени проблеми

Първоначален проблем беше оцветяването на бутоните. Решението беше подходящ метод за това - setStyle.

Имаше проблем с това, че играч можеше да избере вече избрано число и то да се преоцвети в неговия цвят, ако е бил избран преди това от другия или да не стане нищо, освен да стане ред на другия играч, ако е бил избран преди това от същия играч. Този проблем беше решен чрез връщане на булева стойност от сървъра при извикване на метода takeNumber, която ако е false да означава,че числото вече е заето и след това при обработката на събитието се прави проверка на тази стойност, за да не се случва нищо, ако вече е заето.

Друг проблем беше, че при затваряне на прозореца на някой от играчите предварително, тоест преди края на играта, се хвърляше изключение. Решението на този проблем става чрез метода disconnect, който се извиква при затваряне на графичния прозорец на някой играч и чрез проверката дали някой играч не e излязъл от играта в дефиницията на метода takeNumber чрез метода checkIfDisconnected.

Нерешен проблем остава това, че ако един играч затвори прозореца си преди края на играта, другия остане в играта, при включване на нов играч, се стартира нова игра между новия играч и този, който е останал, но оцветените бутони на останалия в играта си остават оцветени. Разчита се на това, че се изпраща съобщение на останалия в играта играч, че другия е излязъл от играта и че при това положение и той ще излезе от играта.

## Източници

1. Използван е като пример имплементацията на играта Tic-Tac-Toe – използвана е идеята за изпращане на съобщения от сървъра на клиентите и callback механизма.
2. Лекцията за RMI приложения – използвана е схемата за създаването на такова приложение стъпка по стъпка.
3. <https://docs.oracle.com> – тук, в документацията за Button в javafx, са намерени необходимите методи за промяна на цвета, размера и формата на бутоните.
4. <https://stackoverflow.com> – тук са търсени идеи за решения на срещнати проблеми.