

Язык Java и разработка Java-приложений

Библиотека **Eclipse Standard Widget Toolkit**

*Разработка интерфейса пользователя
на примере программы «Блокнот настольных игр».*

Романов Владимир Юрьевич,
Московский Государственный Университет им. М.В.Ломоносова
Факультет Вычислительной Математики и Кибернетики
vromanov@cs.msu.su,
vladimir.romanov@gmail.com

Настольные игры

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

<https://www.chessprogramming.org/Games>

CPW Games - Chessprogramming wiki

← → ↻ 🔒 <https://www.chessprogramming.org/Games>

Chess Programming WIKI
CPW

Page **Discussion**

Games

Home * Games

John von Neumann classified Chess as [two-player](#) [zero-sum](#) [abstract strategy](#) chess variants and other games, which are interesting for chess programmers du

[Main page](#)
[Recent changes](#)
[Random page](#)
[Help](#)

Tools

[What links here](#)
[Related changes](#)
[Special pages](#)
[Printable version](#)
[Permanent link](#)
[Page information](#)

A list of recent changes in the wiki [alt-shift-r]

Contents [hide]

- 1 Board Games
 - 1.1 Chess Variants
 - 1.2 Abstract Board Games
 - 1.3 Mancala Games
 - 1.4 Games of Chance
 - 1.5 Without perfect information

Шахматы

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Шахматы
 Шашки
 Сянцзи
 Викинги
 Тамерлан
 Реверси
 Го
 Халма 8x8
 Рендзю

Игроки

Черные

Номо sapience

Незнайка

Белые

Номо sapience

Незнайка

Старт

	A	B	C	D	E	F	G	H	
8									8
7									7
6									6
5									5
4									4
3									3
2									2
1									1
	A	B	C	D	E	F	G	H	

Номо sapience - Номо sapience

1. e2-e4 e7-e5 2. Ng1-f3 Nb8-c6

3. Bf1-b5 a7-a6 4. Bb5-a4

*

Шашки

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Китайские шахматы

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Games Notebook

Шахматы Шашки Сянци Викинги Тамерлан Реверси Го Халма 8x8 Рендзю

Игроки

Черные

Ното сариенсе
Незнайка
Сунь-Цзы
Конфуций

Белые

Ното сариенсе
Незнайка
Сунь-Цзы
Конфуций

Старт

Сунь-Цзы - Конфуций

1. Ec1-e3 Pe7-e6 2. Hb1-d2 Pe6-e5
3. Pe4xe5 Ch8-h4 4. Ch3xh10 Ri10xh10
5. Hd2-e4 Ch4-h2 6. Eg1xh2 Rh10xh2
7. Af1-e2 Rh2xh1 8. Ri1xh1 Cb8-b4
9. Cb3xb10 Ra10xb10 10. Rh1-h10 Cb4-b1
11. Ra1xb1 Rb10xb1 12. Rh10xg10 Rb1xd1
13. Rg10xf10

0-1



Шахматы Тамерлана

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Games Notebook

Шахматы Шашки Сянци Викинги **Тамерлан** Реверси Го Халма 8x8 Рендзю

Игроки

Черные

- Ното сарпенте
- Незнайка
- Тамерлан (Узбек)**
- Тохтамыш (З.Орда)
- Баязид (Турция)
- Насир (Индия)

Белые

- Ното сарпенте**
- Незнайка
- Тамерлан (Узбек)
- Тохтамыш (З.Орда)
- Баязид (Турция)
- Насир (Индия)

Старт

А В С D E F G H I J

10 9 8 7 6 5 4 3 2 1

А В С D E F G H I J

Ното сарпенте - Тамерлан (Узбек)

1. Wf2-g4 e8-e7

*

Реверси

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Шахматы
 Шашки
 Сянци
 Викинги
 Тамерлан
 Реверси
 Го
 Халма 8x8
 Рендзю

Игроки

Черные

Ното сарпенсе

Винни

Сова

Тигра

Белые

Ното сарпенсе

Винни

Сова

Тигра

Старт

Счет

Белые: 18

Черные: 5

	A	B	C	D	E	F	G	H	
8									8
7									7
6									6
5									5
4									4
3									3
2									2
1									1
	A	B	C	D	E	F	G	H	

Сова - Винни

1. d6	c6	2. b6	c4
3. d3	e6	4. f6	d2
5. d1	c2	6. b3	d7
7. e8	a6	8. f5	f7
9. g8	d8	10. c8	g7
11. h7	g5	12. h5	h8
13. g6	f4	14. g4	c5
15. a7	e7	16. f8	b8
17. a5	b5	18. a4	c7
19. b7	c3	20. b4	a2
21. a3	a8	22. Pass	c1
23. b1	h3	24. h4	g3
25. h6	e2	26. h2	e1
27. f1	e3	28. f3	f2
29. g2	h1	30. g1	b2
31. Pass			

0-1

Уголки

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Games Notebook

Шахматы Шашки Сянци Викинги Тамерлан Реверси Го Халма 8x8 Рендзю

Игроки

Черные

Ното сарипсе
Незнайка
Муравьи
Жуки

Белые

Ното сарипсе
Незнайка
Муравьи
Жуки

Старт

Доска

● 8 x 8
● 10 x 10
● 16 x 16

Счет

Белые: 174
Черные: 176

Муравьи - Жуки

1. d10xf10	h2xf2	2. b10xd10	j2xh
3. a7xa5	g1xg3	4. a9xa7	i1xg
5. d10xd8	j4xh4	6. a7xc7	i3xi5
7. b8xb6	g1xe1	8. c9xe9	g2xg
9. c8xe8	i2xg2	10. a6xc6	i4xi6
11. b7xd7	h3xh5	12. d9xf9	h1xh
13. e10xg10	h3xf3	14. f10xf8	g2xe
15. e9xe7	f1xd1	16. c7xc5	g3xe
17. d8xd6	h4xh6	18. f9xf7	f2xd
19. d7xd5	e1xc1	20. e8xg8	i5xg
21. c6xc4	d1xd3	22. e7xg7	i6xg
23. c5xe5	e2xc2	24. f8xh8	h5xf
25. d6xd4	d2xb2	26. f7xh7	e3xc
27. c4xe4	h6xf6	28. g8xi8	c2xc
29. d4xf4	f5xf7	30. d5xf5	g6xg
31. g7xi7	g4xg6	32. e5xe3	g5xg
33. h8xh6	d3xb3	34. h7xh5	f3xd
35. e4xg4	g6xe6	36. i8xi6	c3xc

1-0

Практикум по языку Java

Возможные темы практикума

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- Реализация **правил игры** для новой настольной игры (например, с сайта <https://www.chessprogramming.org/Games>)
 - Японские шахматы
 - ...
- Реализация **новых алгоритмов игры** для уже существующих игр (**Незнайка, Винни Пух, Сова, ...**)
- Реализация интерфейса пользователя **с помощью новой библиотеки**
 - **Swing**
 - **JavaFX**
 - **Google Web Toolkit** (клиент и сервер оба написаны на Java)
- **Расширение интерфейса пользователя**
 - Организация соревнований между алгоритмами (матчи, турниры, ...)
 - Хранение архивов партий и соревнований в БД (реляционных (JDBC),...)
 - Редактор начальных позиций игр

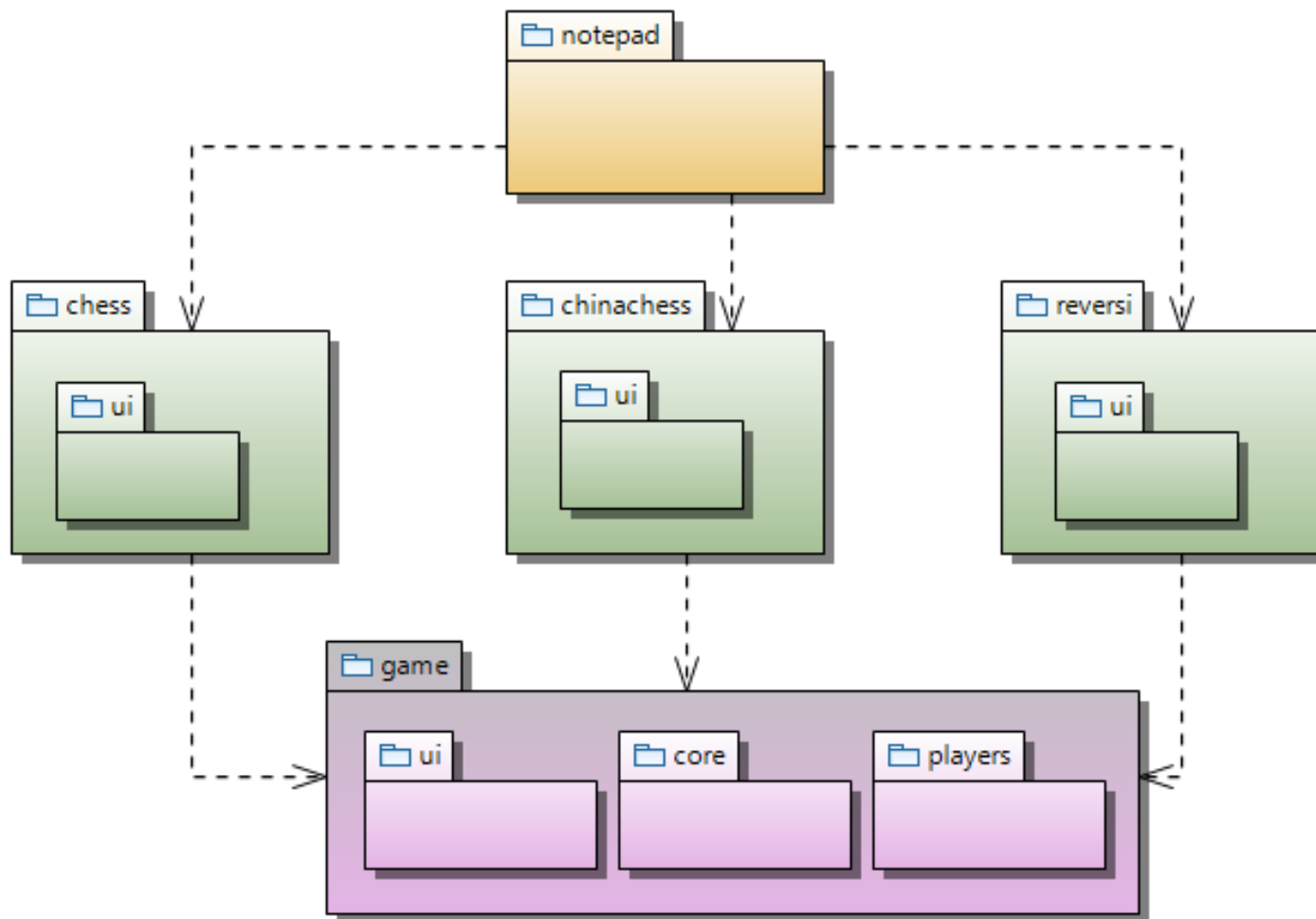
Проектирование архитектуры.

Выделение уровней в настольных играх

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Идентификация *пакетов* верхнего уровня

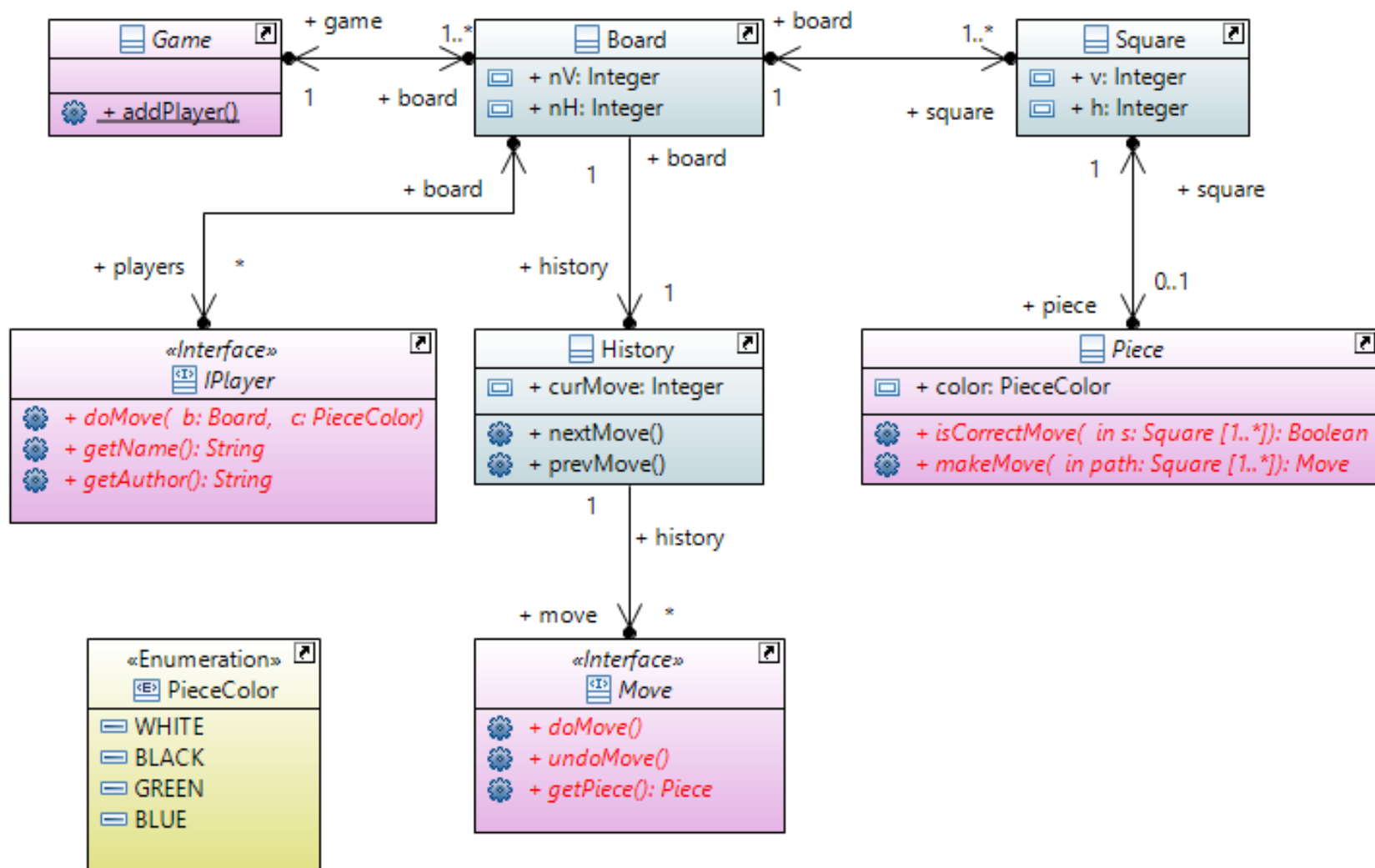


Проектирование архитектуры.

Ядро настольных игр

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

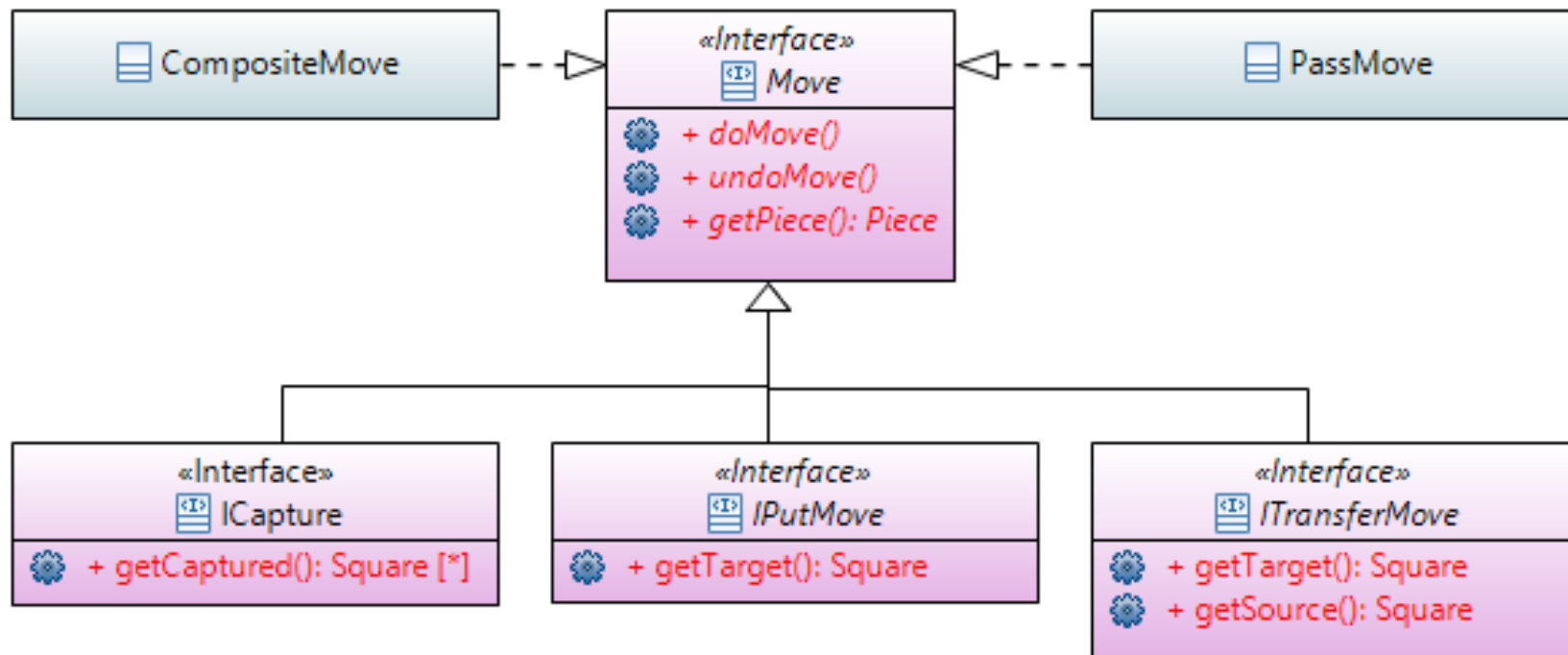


Проектирование архитектуры.

Ядро настольных игр. Ходы настольных игр

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

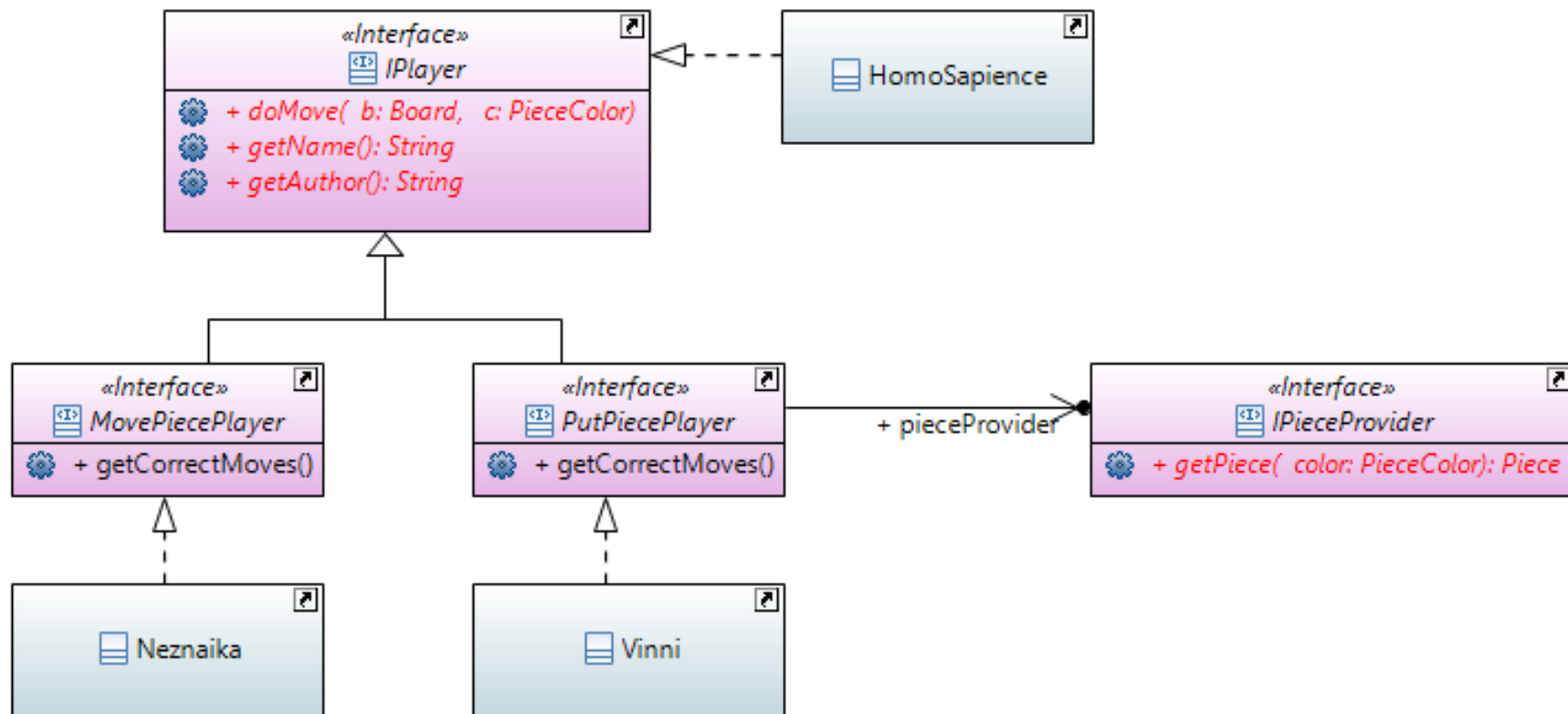


Проектирование архитектуры.

Ядро настольных игр. Игроки настольных игр

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



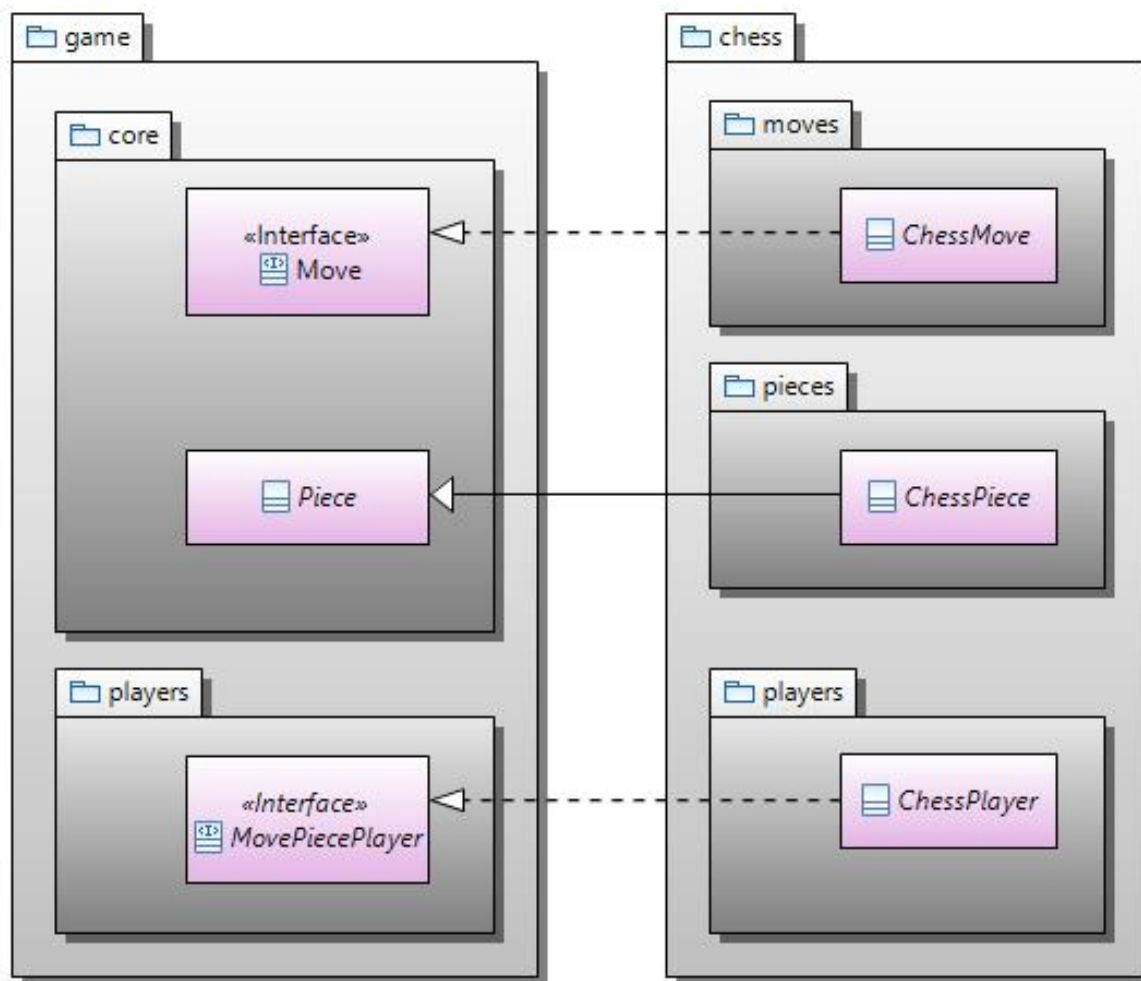
Проектирование архитектуры

Зависимости между пакетами.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Идентификация зависимостей между пакетами.

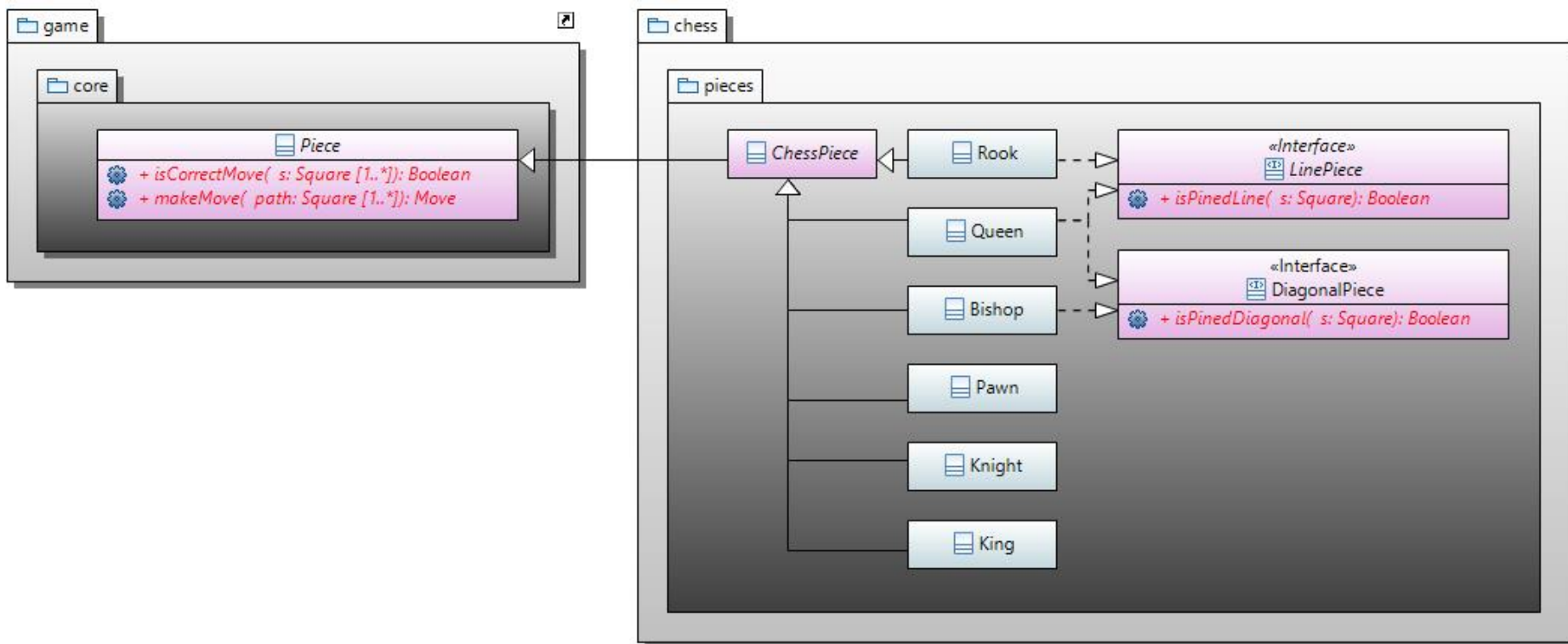


Проектирование классов

Классы-фигуры в шахматной программе

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

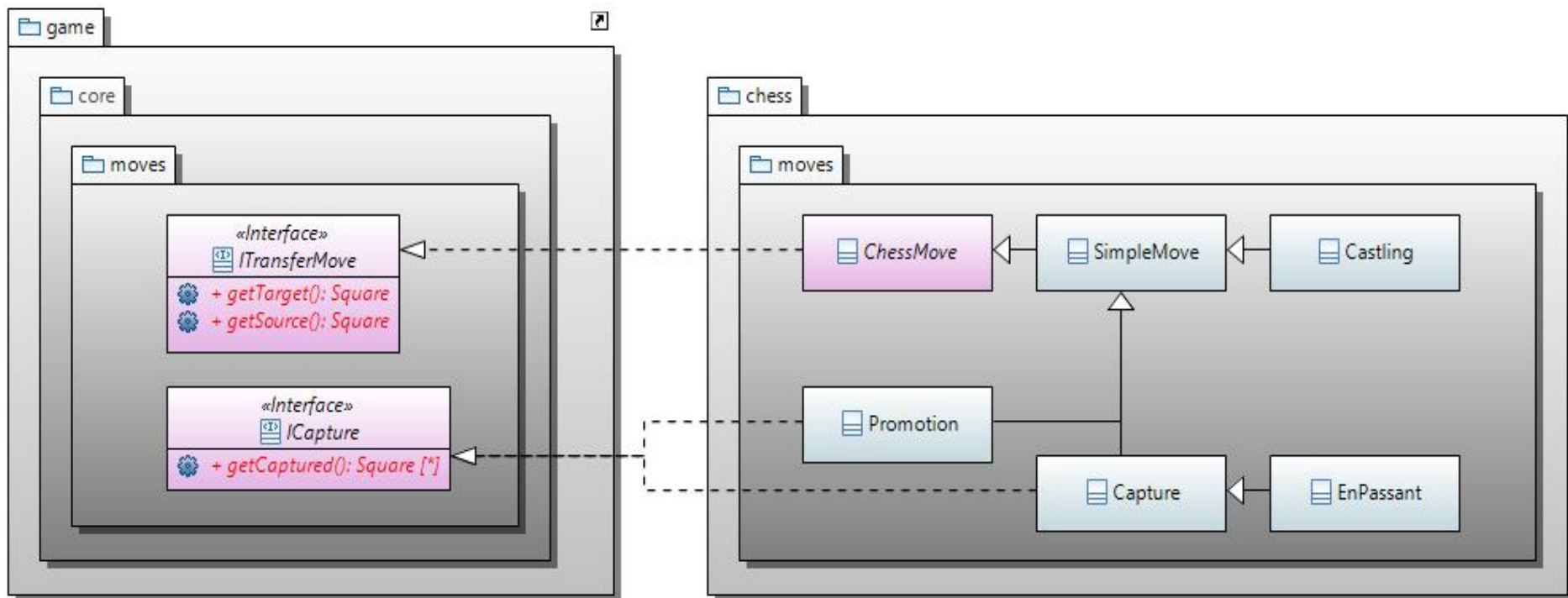


Проектирование классов

Классы-ходы в шахматной программе

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

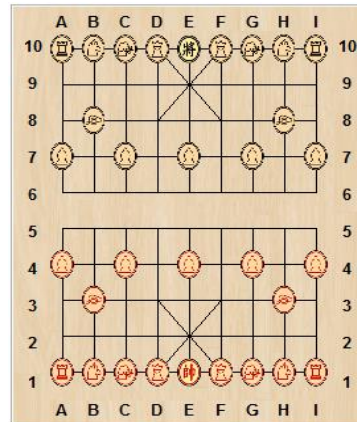
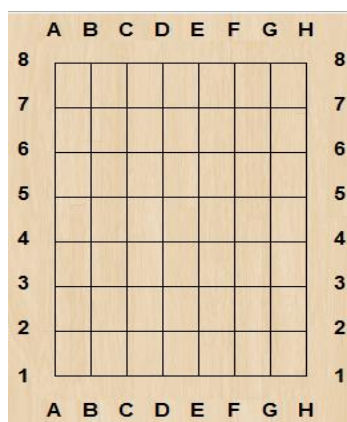
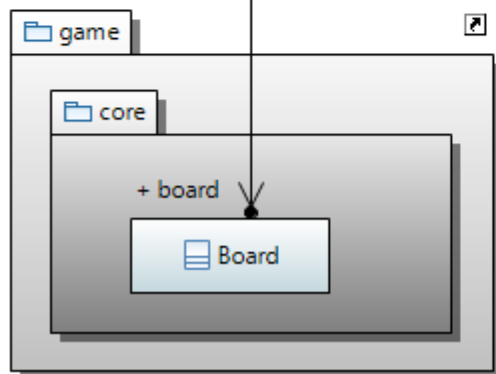
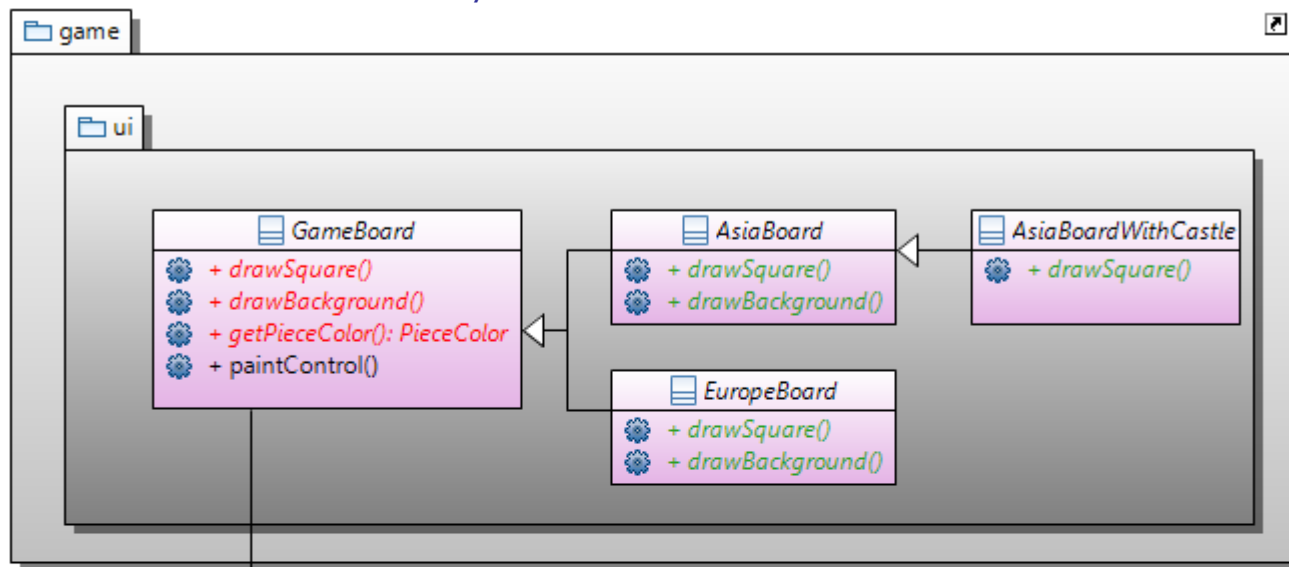


Проектирование интерфейса пользователя

Интерфейс пользователя для игры на доске.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

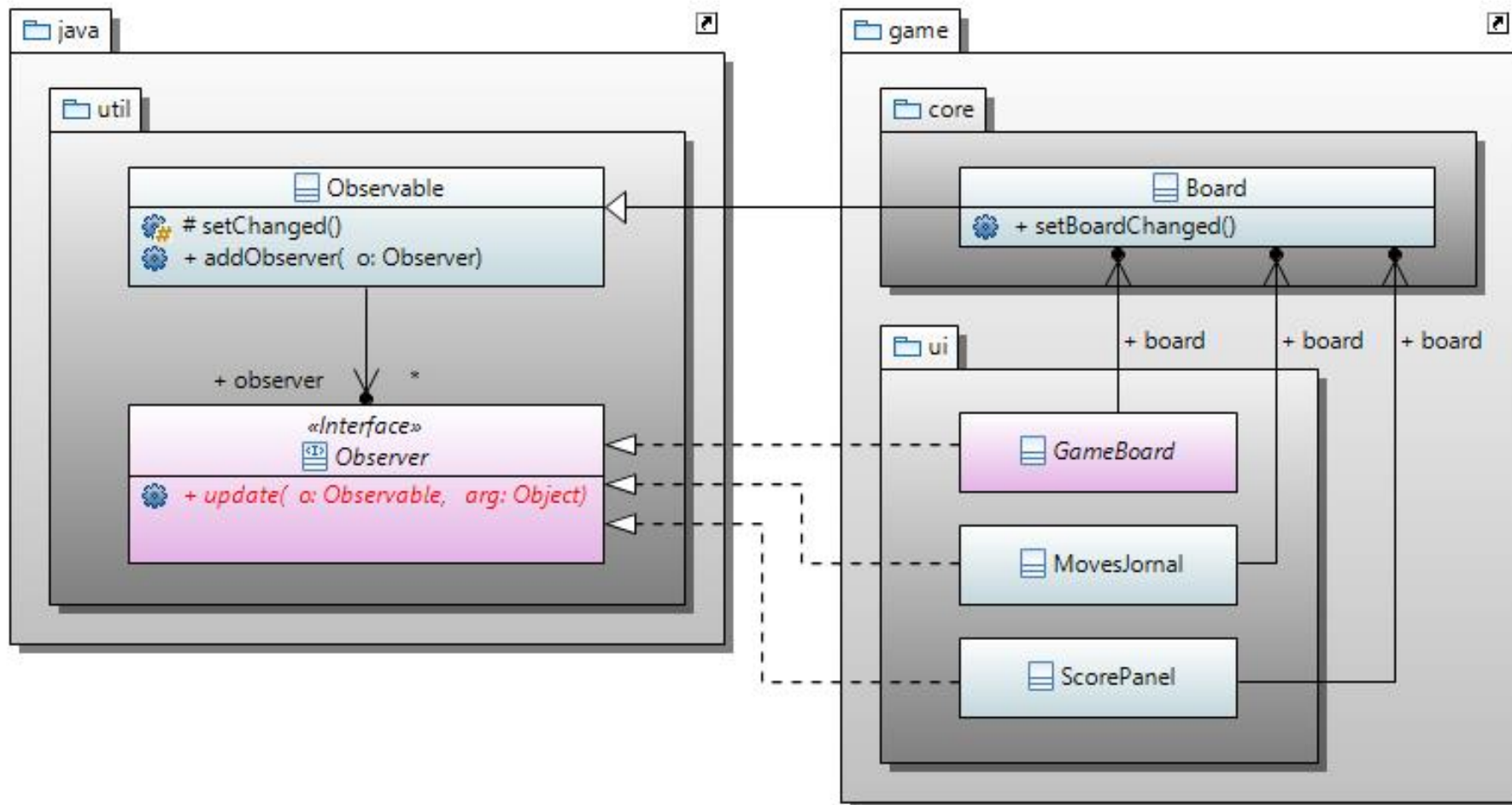


Проектирование интерфейса пользователя

Обозреваемый (доска) и обозреватели

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

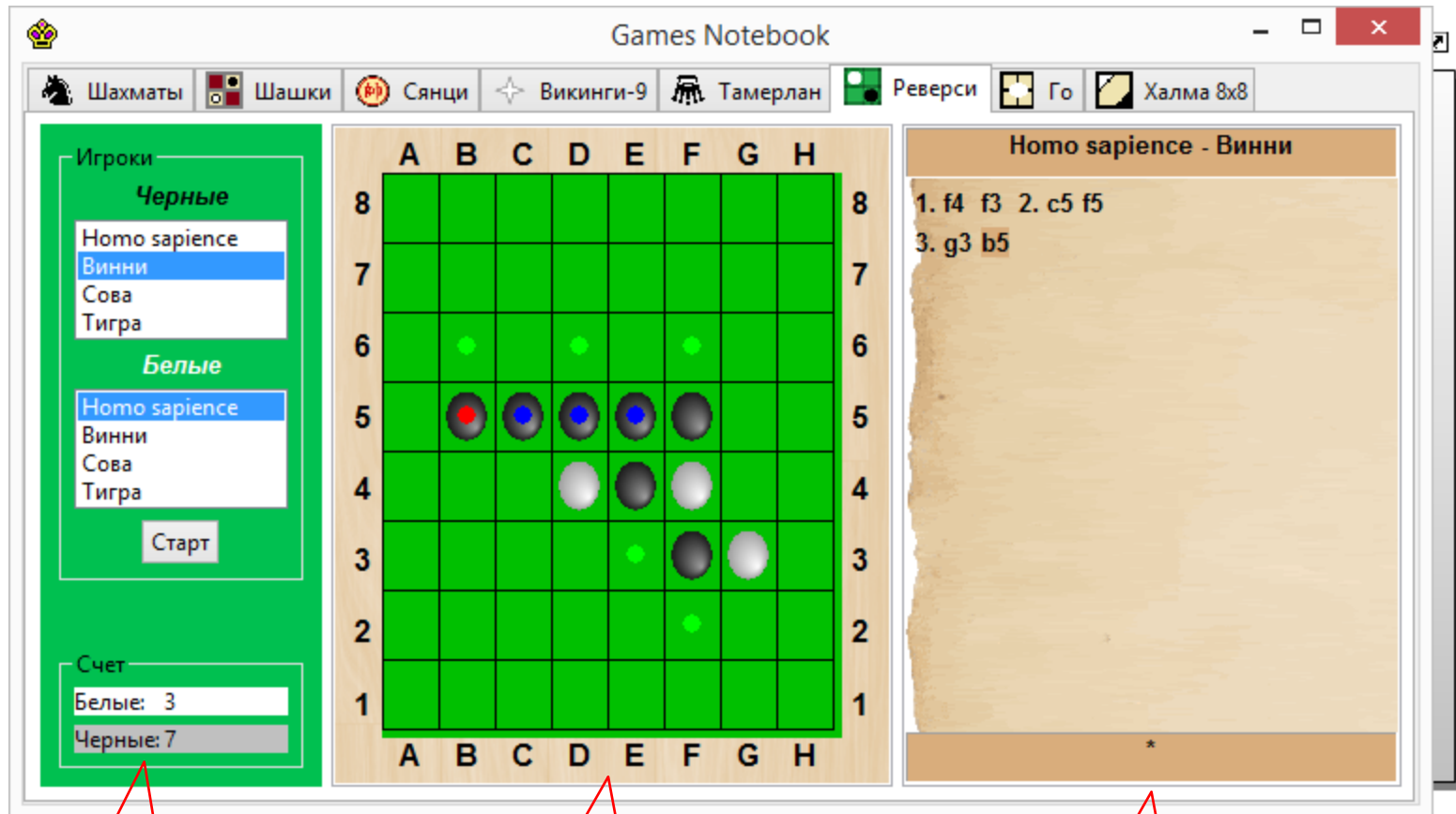


Проектирование интерфейса пользователя

Обозреватели доски

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



ScorePanel

GameBoard

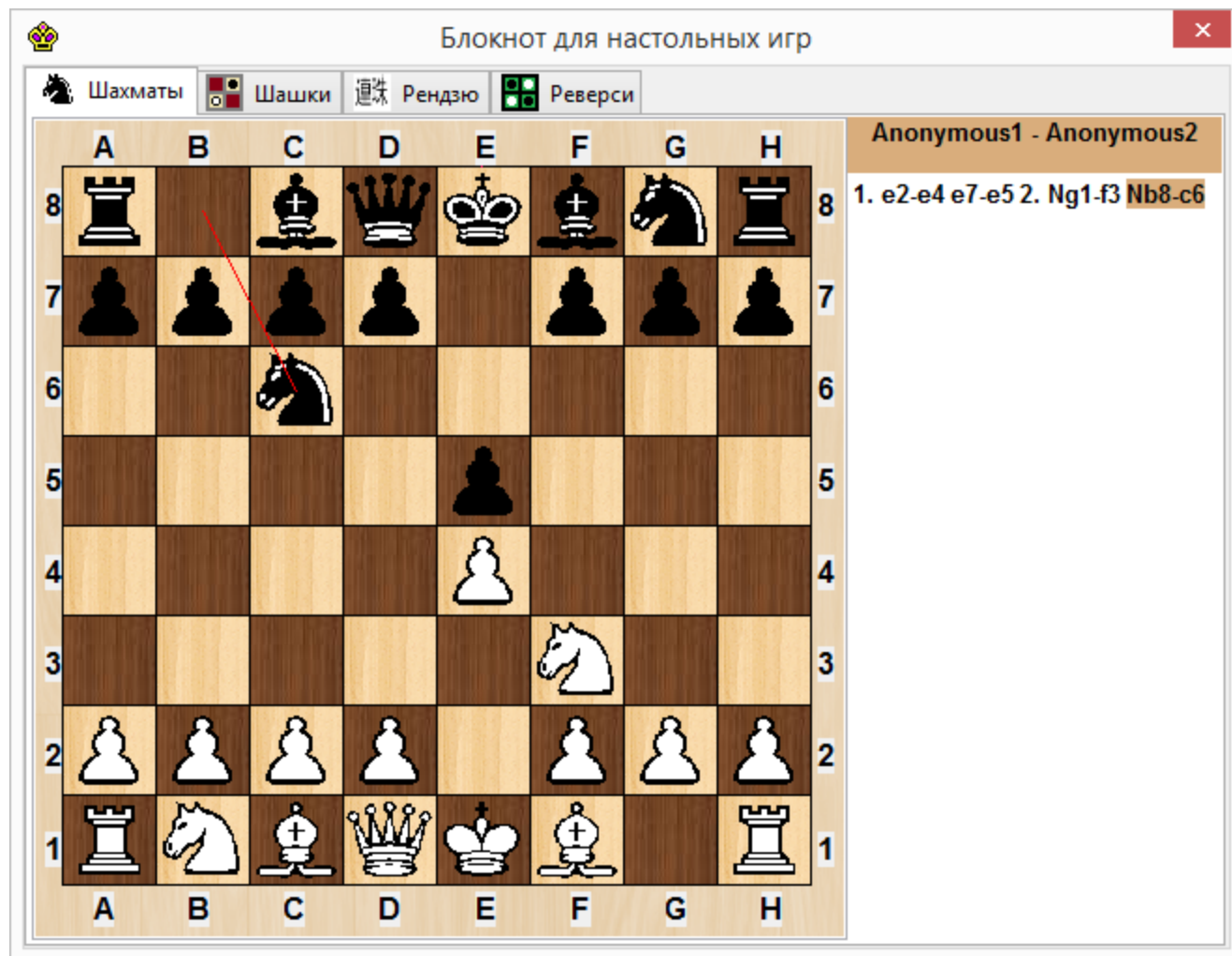
MovesJornal

Блокнот игр. Шахматы.

Подсветка последнего хода

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

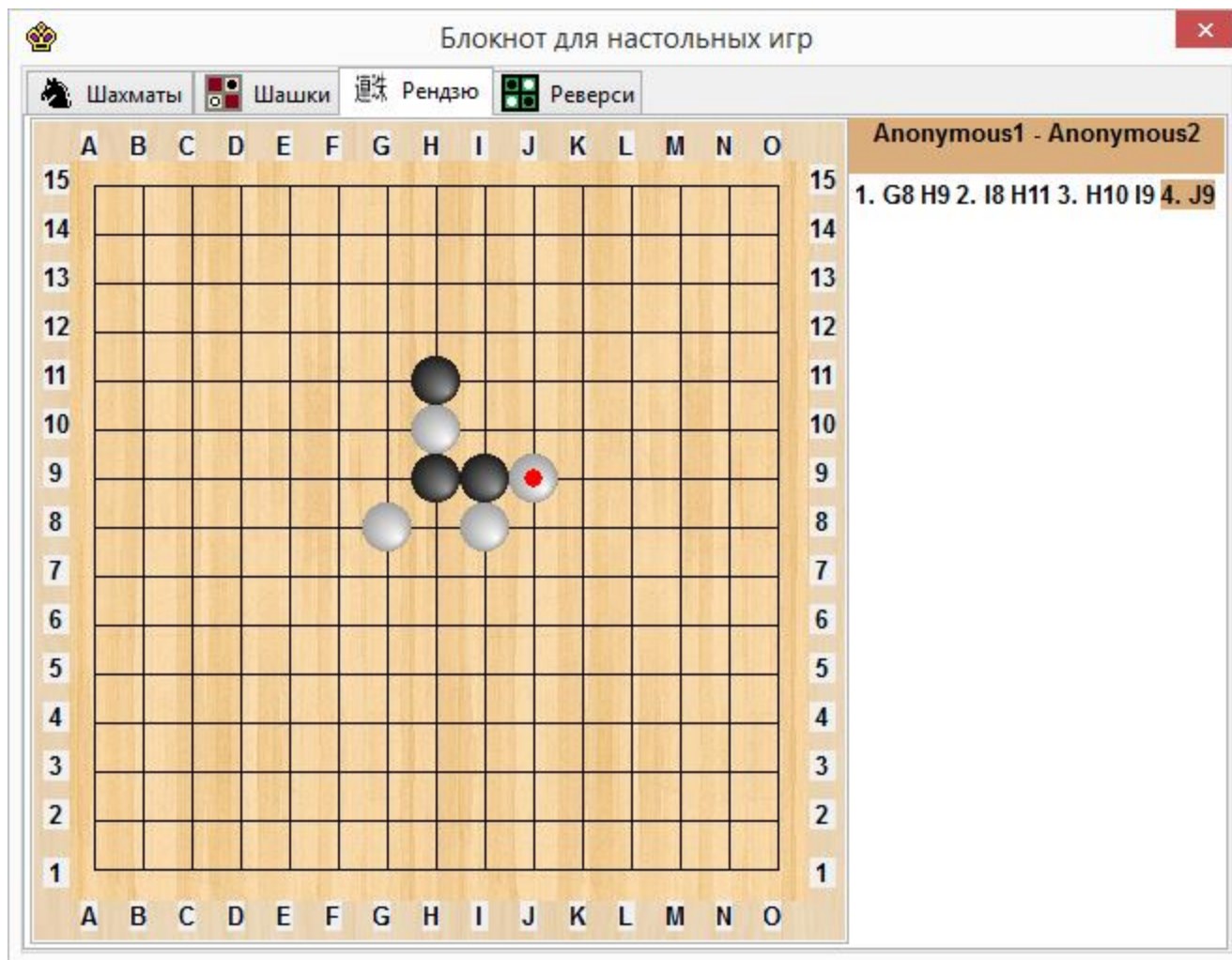


Блокнот игр. Рендзю.

Подсветка последнего хода

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

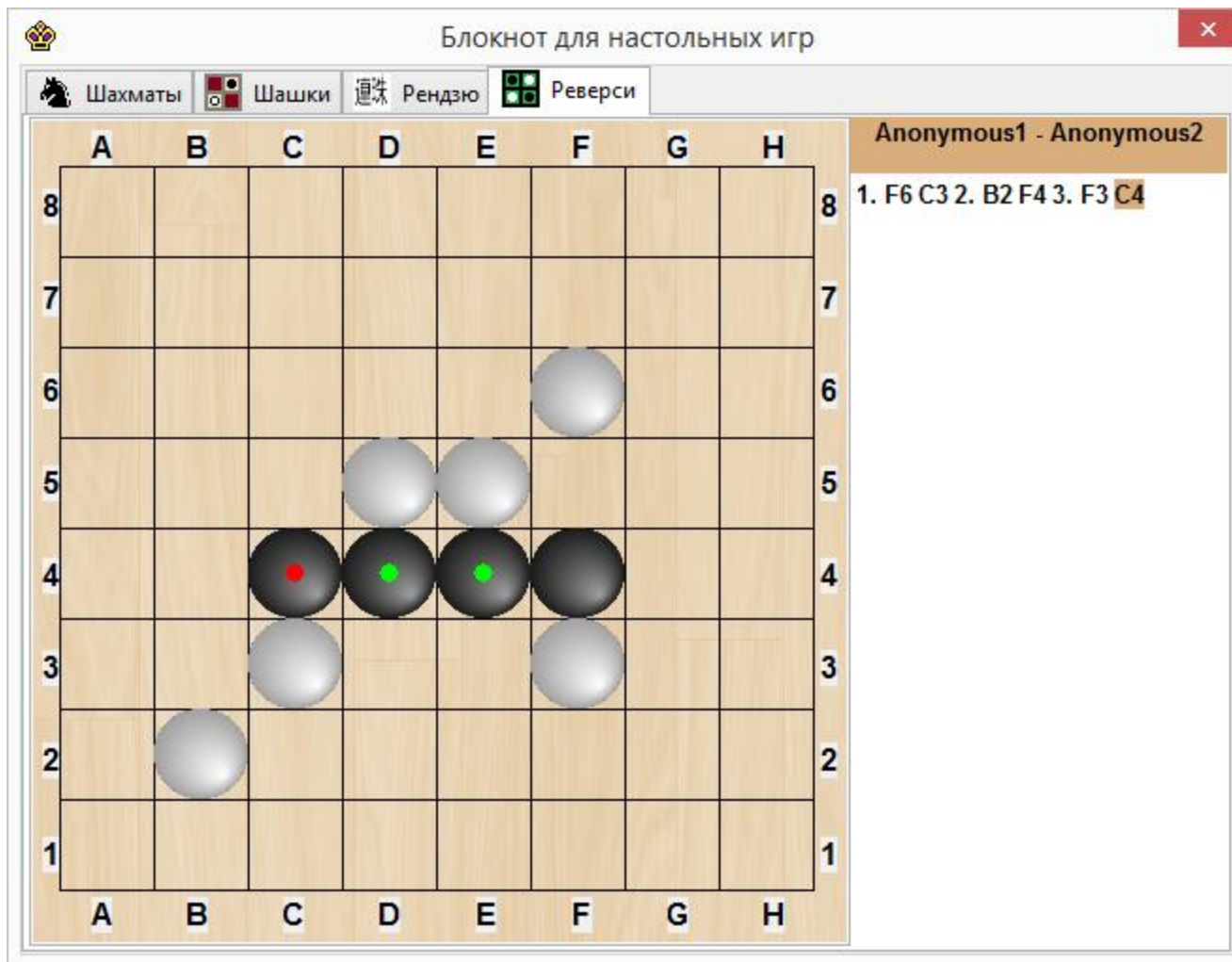


Блокнот игр. Реверси.

Подсветка последнего хода

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Назначение SWT

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- **Standard Widget Toolkit:** Инструмент для разработки интерфейса пользователя
 - ❑ Разработан в **IBM**
 - ❑ Используется в проектах **www.eclipse.org**
 - ❑ Эффективный
 - ❑ Переносимый
 - ❑ Низкий уровень реализации: есть доступ к возможностям операционной системы
- Сайт проекта: **eclipse.org/swt**

Элементы библиотеки SWT

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Сайт проекта: <https://www.eclipse.org/swt/widgets/>

SWT Widgets

Below are screenshots and links to documentation for many of the widgets included in SWT. For a complete list of classes including those that don't screenshot well, see the SWT Javadoc.



Browser
javadoc - snippets



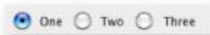
Button (SWT.ARROW)
javadoc - snippets



Button (SWT.CHECK)
javadoc - snippets



Button (SWT.PUSH)
javadoc - snippets



Button (SWT.RADIO)
javadoc - snippets



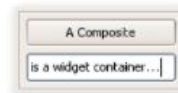
Button (SWT.TOGGLE)
javadoc - snippets



Canvas
javadoc - snippets



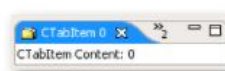
Combo
javadoc - snippets



Composite
javadoc - snippets



CoolBar
javadoc - snippets



CTabFolder
javadoc - snippets



DateTime
javadoc - snippets



Фрагменты кода для использования элементов библиотеки SWT

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Сайт проекта: <https://www.eclipse.org/swt/snippets/>

SWT Snippets

Snippets are minimal stand-alone programs that demonstrate specific techniques or functionality. Often a small example is the easiest way to understand how to use a particular feature. (If you are looking for large examples, like *ControlExample*, see the [SWT Examples](#), and if you are programming with JFace, you may find these [JFace Snippets](#) useful).

Snippets also help isolate problems. The best way to report an SWT bug is to write your own snippet showing the problem and paste it into the bug report. For a snippet template, see the "Hello World" example.

Note that the examples here are often edited for brevity rather than completeness. They are intended to guide the reader towards the correct solution, rather than be finished products. These snippets are tested against the HEAD stream and may sometimes reference new API or require bug fixes from there.

To run a snippet, simply **import SWT into your Eclipse workspace**, create a new Java project that depends on SWT, copy the desired snippet to the clipboard, and paste it into a new snippet class. (If you are using eclipse 3.2 M1 or earlier, you need to create the class using the *New Class* wizard before pasting; but since 3.2 M1 you can simply select your project and paste, and the class is created for you). Run by selecting the class and then selecting **Run > Run As > Java Application**.

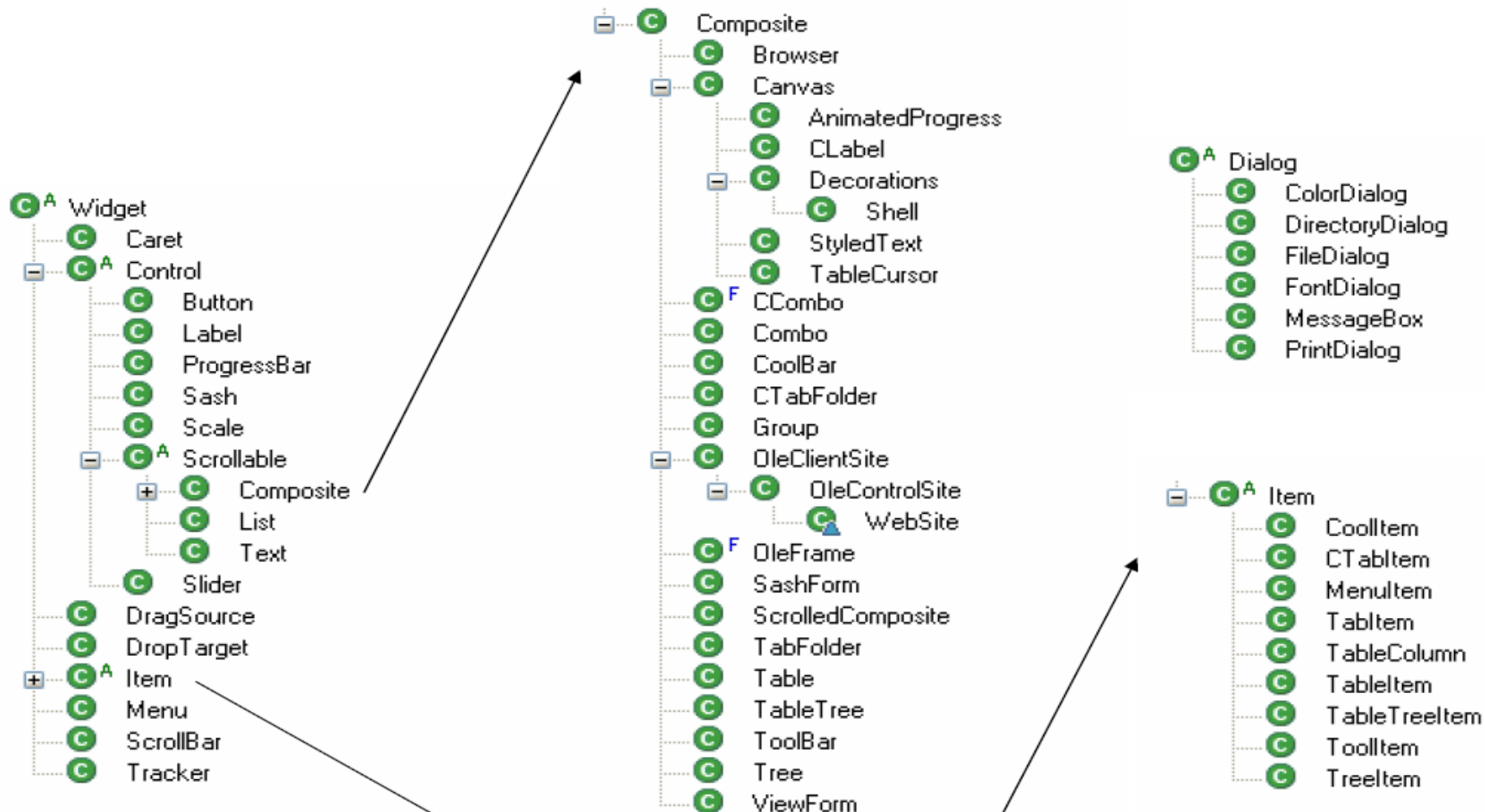
To contribute a new snippet, [create a snippet contribution report in Bugzilla](#). Thanks in advance for your contribution!

- "Hello World"
 - "Hello World"
- Accessibility
 - using an accessible listener to provide state information – (preview)
 - provide text that will be spoken for an image button – (preview)
 - give accessible names to a tree and its tree items – (preview)
 - respond to text-based questions from an AT – (preview)
 - tell a screen reader about updates to a non-focused descriptive area – (preview)
 - use accessible relations to provide additional information to an AT – (preview)
 - provide a way for an AT to set text attributes in a StyledText – (preview)
 - declare a message area to be a "live region" – (preview)
- Browser
 - check if the browser is available or not – (preview)
 - bring up a browser (single window) – (preview)
 - bring up a browser with non-in blocker – (preview)

Иерархия управляющих элементов (widgets).

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



SWT. Основные классы библиотеки:

Класс **Display**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- Представляет *рабочее место (workstation)* мониторы, клавиатуру, мышку
- Отвечает за распределение событий в *цикле событий (event loop)*
- Содержит список *окон верхнего уровня (Shells)*
- Содержит список *мониторов (Monitors)*

SWT. Основные классы библиотеки:

Класс **Shell**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

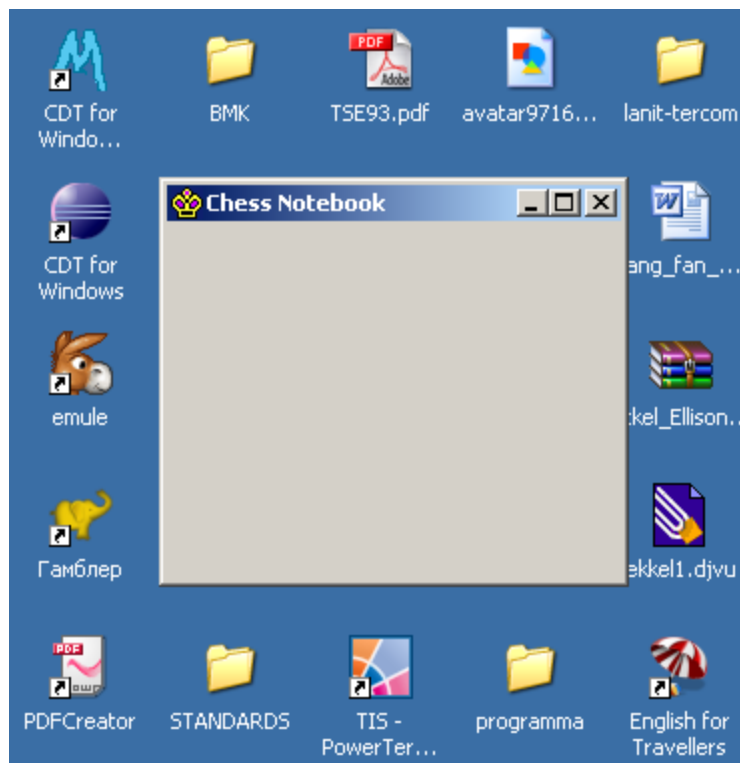
- Представляет *окно* на экране
- Это корень дерева состоящего из:
 - ▣ *Composites* (составной элемент)
 - ▣ *Controls* (управляющий элемент)
- **Shell** – потомок класса **Widget**

SWT. Вид класса **Shell** на экране.

Окно с заголовком *Chess Notebook*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



SWT. Основные классы библиотеки:

Класс **Composite**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- **Composite** (составной элемент) - управляющий элемент (control) который может состоять из других составных элементов и управляющий элементов
- **Composite** — потомок класса **Widget**

SWT. Основные классы библиотеки:

Класс **Control**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- **Control** (управляющий элемент) – представляет *легковесный (heavyweight)* элемент операционной системы
- Примеры управляющих элементов: Button, Label, Text, Tree, Shell, Composite, ...
- **Control** – потомок класса **Widget**

SWT. Цикл событий (event loop)

- В **SWT**, цикл событий должен быть явно закодирован в приложении
- Цикл событий постоянно *читает и распределяет события* интерфейса пользователя поступающие из операционной системы и «отдает» CPU когда событий нет.
- Цикл событий завершается когда завершается приложение. Обычно когда закрывается окно.

SWT. Пример цикла событий

```
while (!shell.isDisposed()) {  
  
    if (!display.readAndDispatch())  
        display.sleep ();  
  
}
```


Шахматы. Минимальное «приложение»

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
import org.eclipse.swt.widgets.*;

public class Chess {
    public static void main(String[] args) {
        final Display display = new Display();

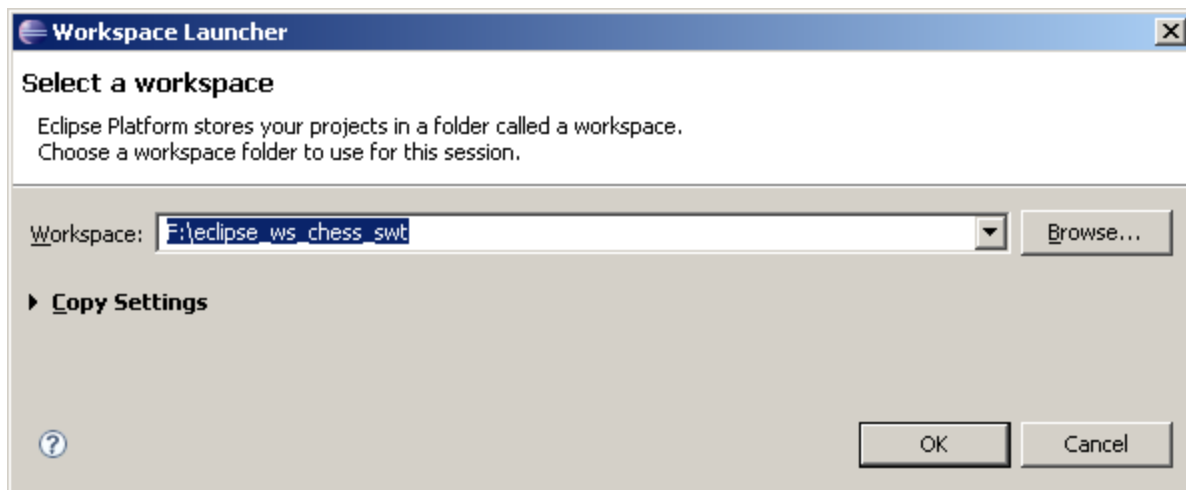
        final Shell shell = new Shell(display);
        shell.setSize(600, 500);
        shell.setText("Chess Notebook");
        shell.open();
        while (!shell.isDisposed()) {
            if (!display.readAndDispatch())
                display.sleep();
        }
        display.dispose();
    } // main
} // class Chess
```

Шахматы. Создание рабочего места

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

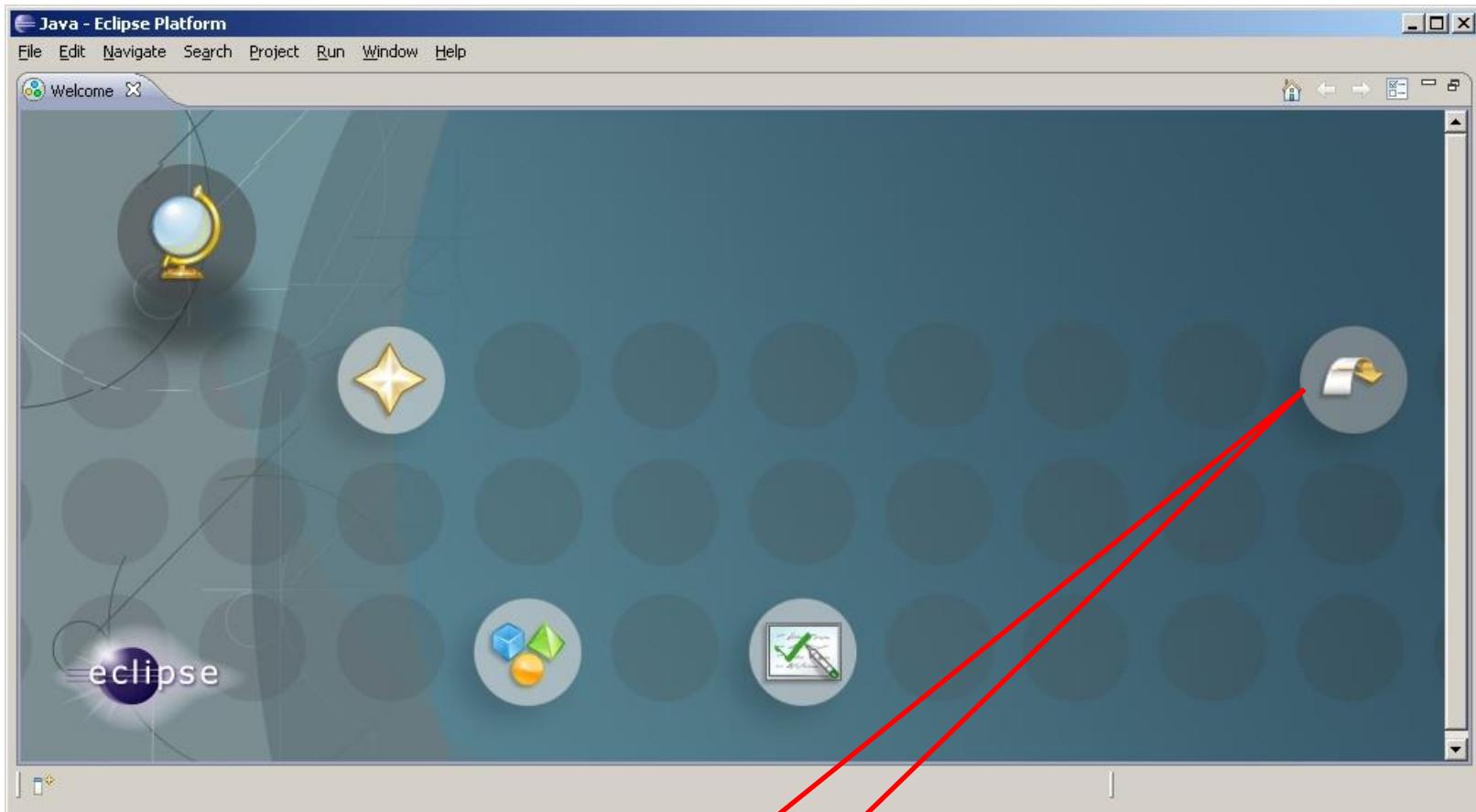
■ Files | Switch Workspace | Other ...



Шахматы. Рабочее место. Wellcome

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

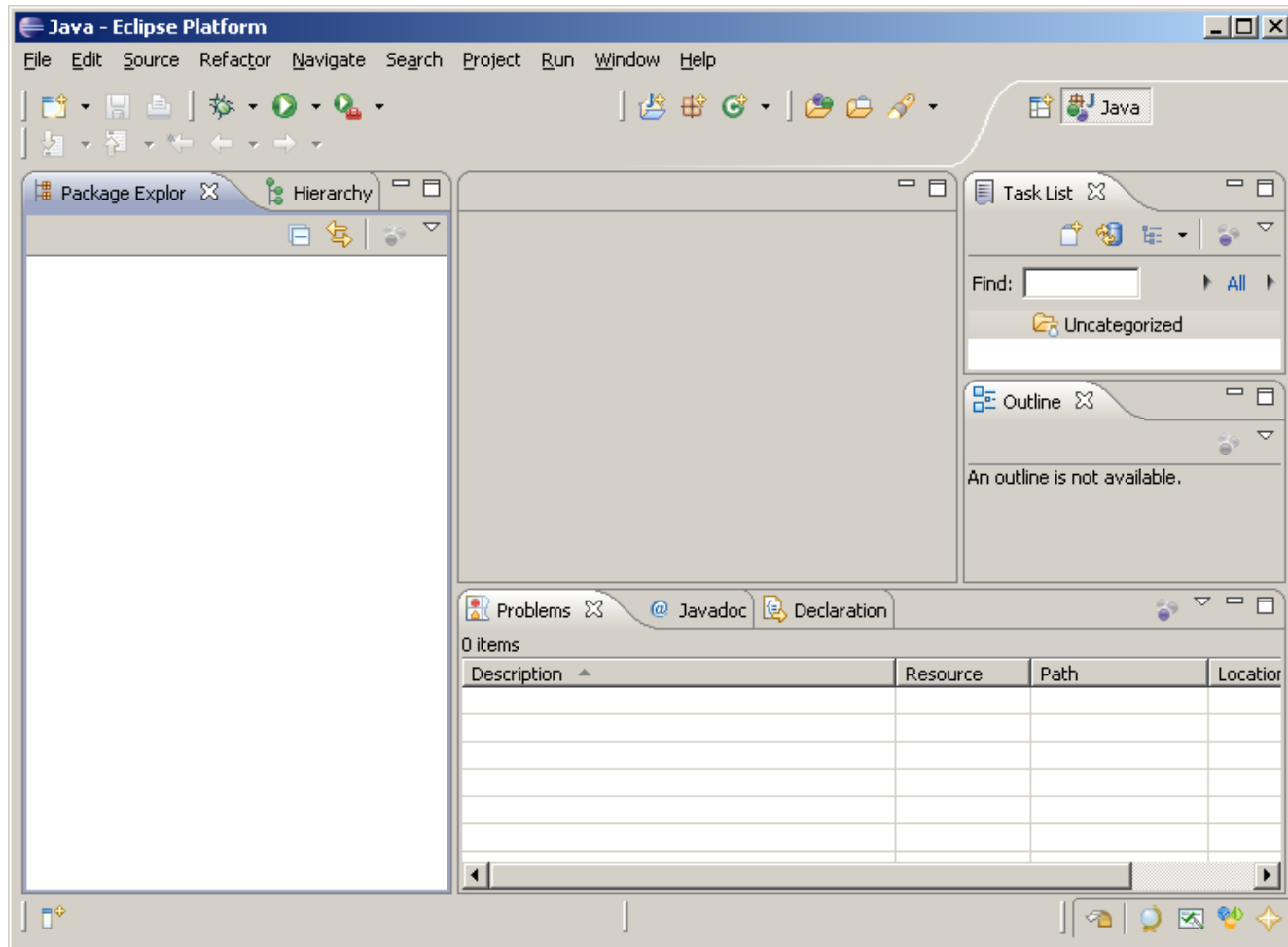


Нажать здесь

Шахматы. Workbench (Станок ??)

МГУ им. М.В.Ломоносова. Факультет ВМК.

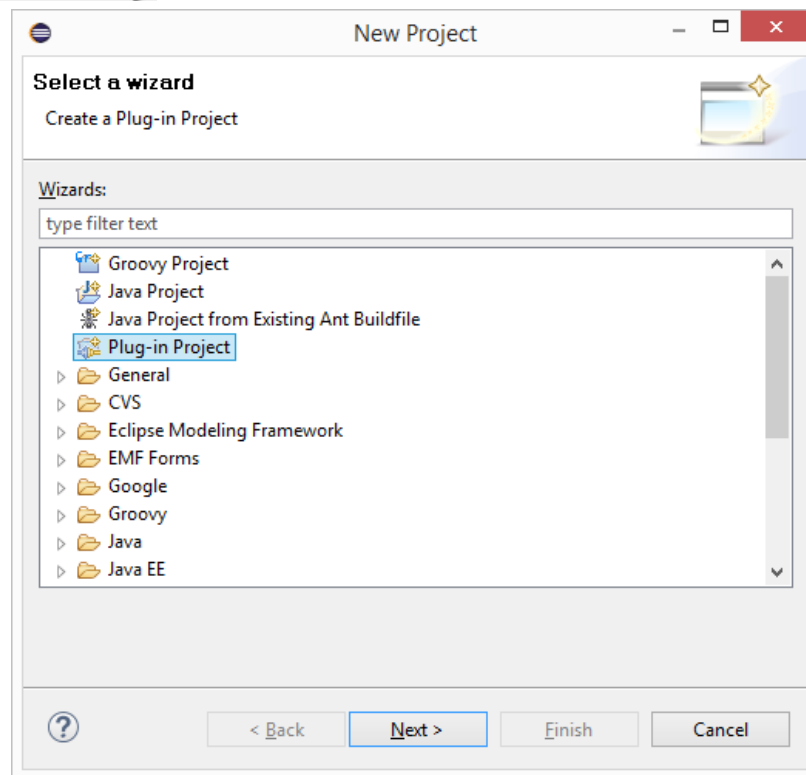
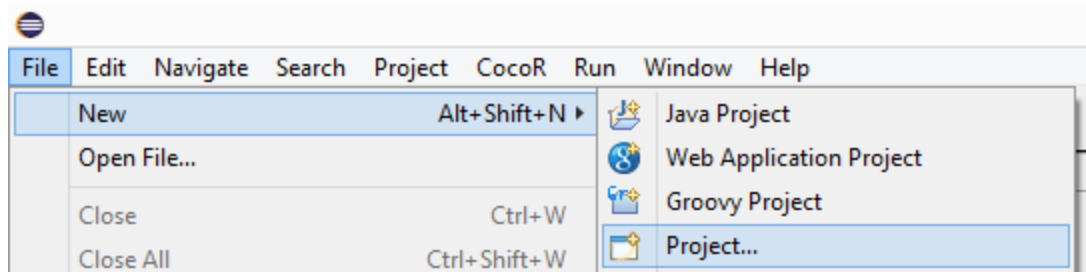
Романов Владимир Юрьевич ©2025



Шахматы. Создание нового проекта

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Шахматы. Создание проекта

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Ввести имя проекта
ChessSWT

New Plug-in Project

Plug-in Project
Create a new plug-in project

Project name: ChessSWT

☒ Use default location
Location: C:\ws-java2017draft\ChessSWT Browse...

Project Settings
☒ Create a Java project
Source folder: src
Output folder: bin

Target Platform
This plug-in is targeted to run with:
☒ Eclipse version: 3.5 or greater
☐ an OSGi framework: Equinox

Working sets
☐ Add project to working sets
Working sets: Select...

? < Back Next > Finish Cancel

Шахматы. Создание проекта

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

New Plug-in Project

Content
Enter the data required to generate the plug-in.

Properties

ID:

Version:

Name:

Vendor:

Execution Environment:

Options

☒ Generate an activator, a Java class that controls the plug-in's life cycle
Activator:

☒ This plug-in will make contributions to the UI

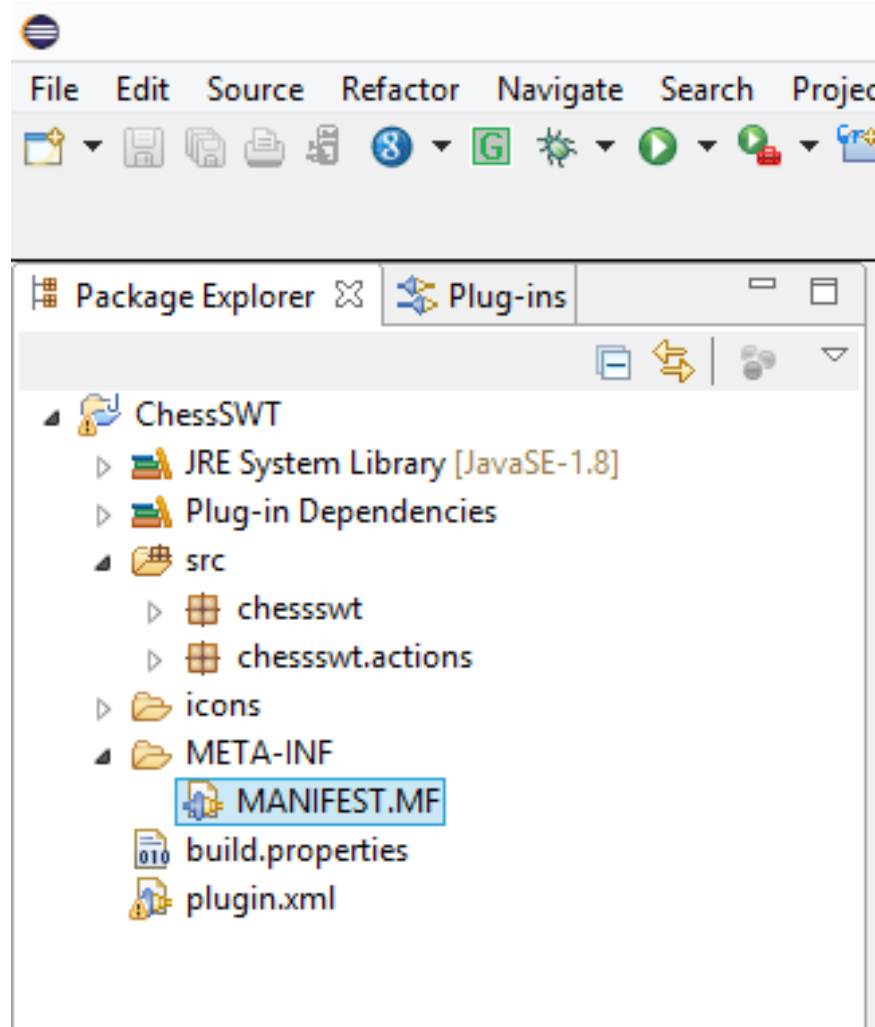
☐ Enable API analysis

Rich Client Application
Would you like to create a 3.x rich client application? ☐ Yes ☒ No

Шахматы. Создание проекта

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Шахматы. Создание “главного” класса

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

■ Files | New | Class ...

Ввести имя класса
Chess

Попросить создать
метод **main()**

Java Class

The use of the default package is discouraged.

Source folder: ChessSWT/src Browse...

Package: (default) Browse...

☐ Enclosing type: Browse...

Name: Chess

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add... Remove

Which method stubs would you like to create?

☒ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

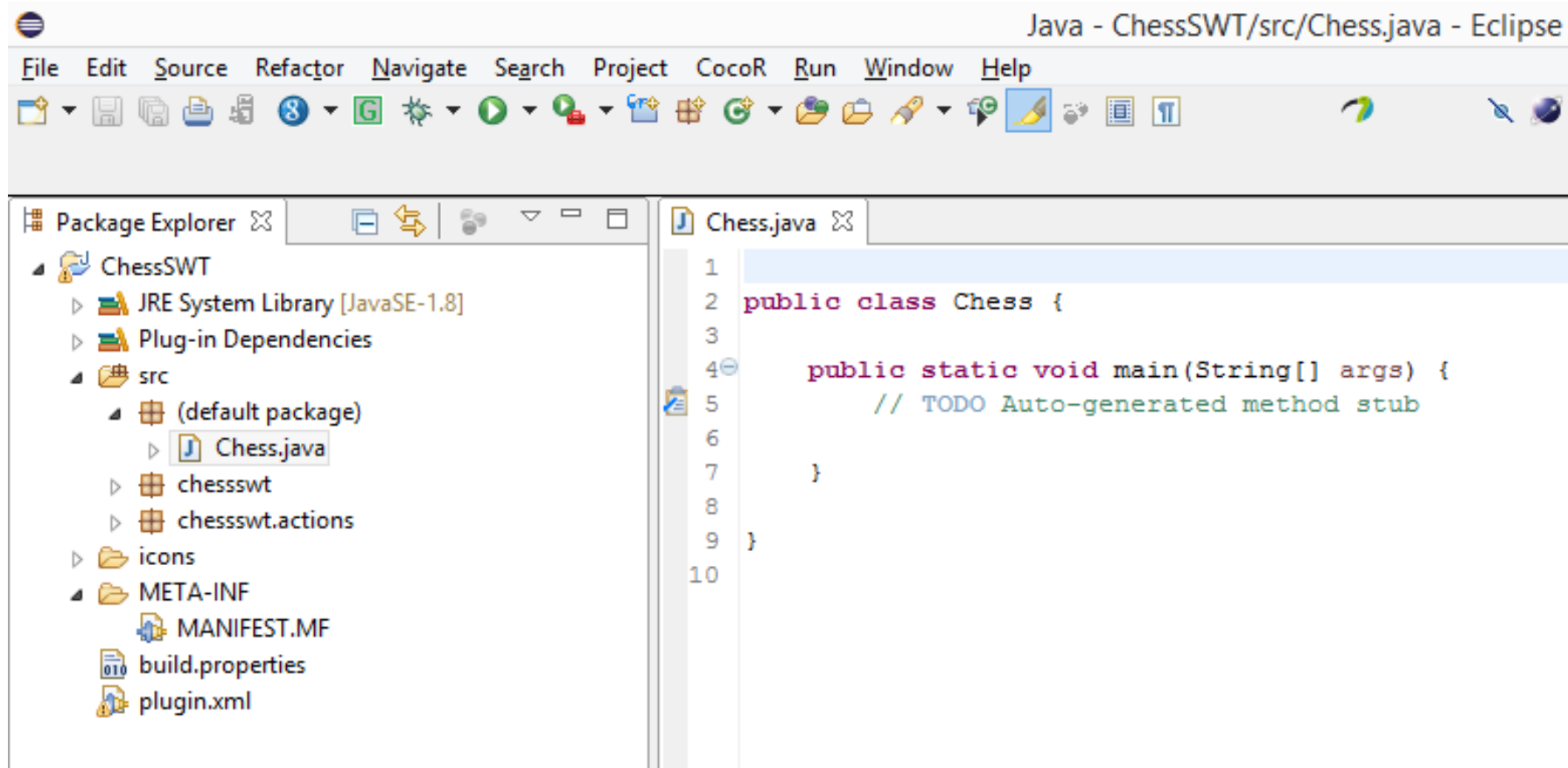
Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

Finish Cancel

Шахматы. Сгенерированный «главный» класс Chess

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Шахматы. Минимальное приложение

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
import org.eclipse.swt.widgets.*;

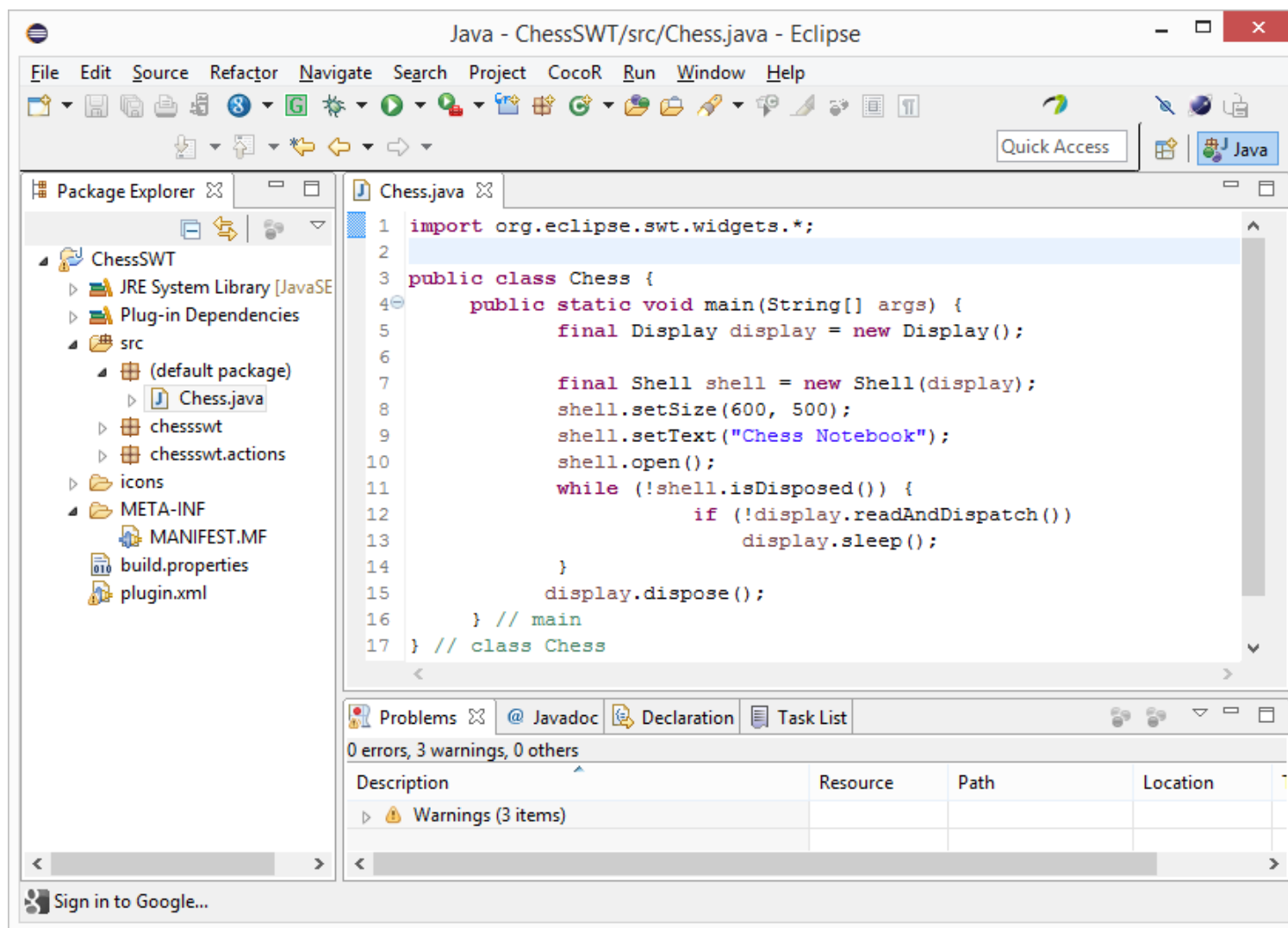
public class Chess {
    public static void main(String[] args) {
        final Display display = new Display();

        final Shell shell = new Shell(display);
        shell.setSize(600, 500);
        shell.setText("Chess Notebook");
        shell.open();
        while (!shell.isDisposed()) {
            if (!display.readAndDispatch())
                display.sleep();
        }
        display.dispose();
    } // main
} // class Chess
```

Шахматы. «Минимальные» шахматы в окне редактора

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Шахматы. Горячие клавиши редактора

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Ctrl + Shift + L

показать список всех «горячих клавиш»

Activate Editor	F12
Activate Task	Ctrl+F9
Add Artifact to Target Platform	Ctrl+Alt+Shift+A
Add Javadoc Comment	Alt+Shift+J
All Instances	Ctrl+Shift+N
Backward History	Alt+Left
Build All	Ctrl+B
Change Method Signature	Alt+Shift+C
Checkin	Ctrl+6
Close	Ctrl+F4
Close All	Ctrl+Shift+F4
CocoR Command	Ctrl+6
Collapse All	Ctrl+Shift+Numpad
Commit...	Ctrl+#

Press "Ctrl+Shift+L" to open the preference page.

Шахматы. Открытие исходных текстов горячей клавишей F3

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

F3

```
Chess.java
1 import org.eclipse.swt.widgets.*;
2
3 public class Chess {
4     public static void main(String[] args) {
5         final Display display = new Display();
6         final Shell shell = new Shell(display);
7         shell.setSize(600, 500);
8         shell.setText("Chess Notebook");
9         shell.open();
10         while (!shell.isDisposed()) {
11             if (!display.readAndDispatch())
12                 display.sleep();
13         }
14         display.dispose();
15     } // main
16 } // class Chess
```

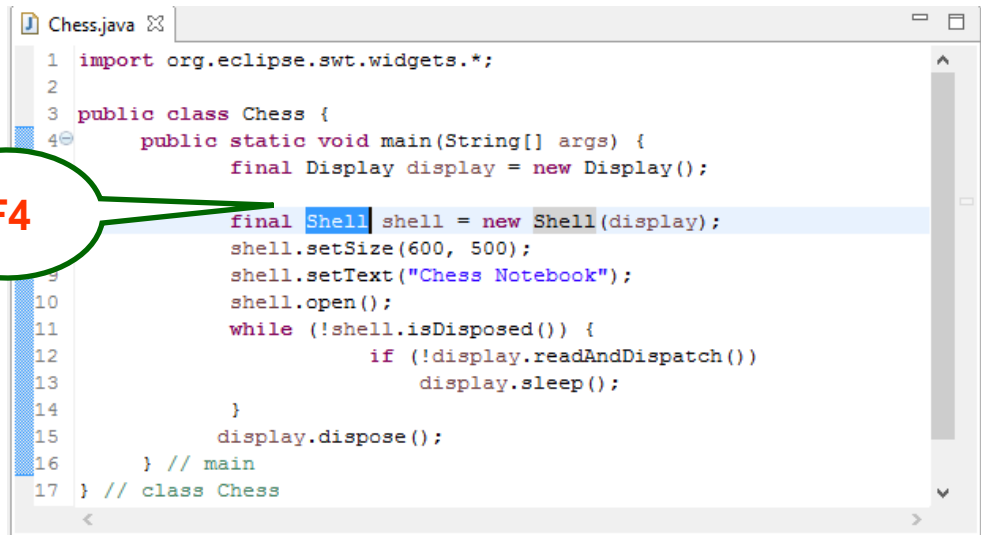


```
Chess.java  Shell.class
114 * @see <a href="http://www.eclipse.org/swt/snippets/#shell">Shell
115 * @see <a href="http://www.eclipse.org/swt/examples.php">SWT Exam
116 * @see <a href="http://www.eclipse.org/swt/">Sample code and fur
117 * @noextend This class is not intended to be subclassed by client
118 */
119 public class Shell extends Decorations {
120     Menu activeMenu;
121     Tooltip [] toolTips;
122     long /*int*/ hIMC, hwndMDIClient, lpstrTip, tooltipHandle, ba
123     int minWidth = SWT.DEFAULT, minHeight = SWT.DEFAULT;
124     long /*int*/ [] brushes;
125     boolean showWithParent, fullScreen, wasMaximized, modified, co
126     String toolTitle, balloonTitle;
127     long /*int*/ toolIcon, balloonIcon;
128     long /*int*/ windowProc;
129     Control lastActive;
130     SHACTIVATEINFO psai;
```


Шахматы. Открытие иерархии наследования горячей клавишей F4

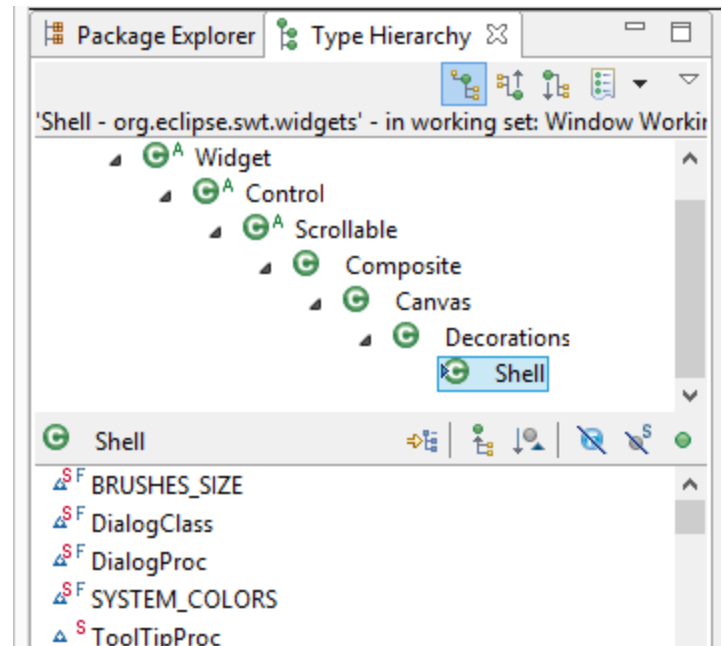
МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



The image shows a snippet of Java code in the Eclipse IDE. A green callout bubble with the text 'F4' points to the 'Shell' class in the line 'final Shell shell = new Shell(display);'. The code is as follows:

```
1 import org.eclipse.swt.widgets.*;
2
3 public class Chess {
4     public static void main(String[] args) {
5         final Display display = new Display();
6         final Shell shell = new Shell(display);
7         shell.setSize(600, 500);
8         shell.setText("Chess Notebook");
9         shell.open();
10        while (!shell.isDisposed()) {
11            if (!display.readAndDispatch())
12                display.sleep();
13        }
14        display.dispose();
15    } // main
16 } // class Chess
```

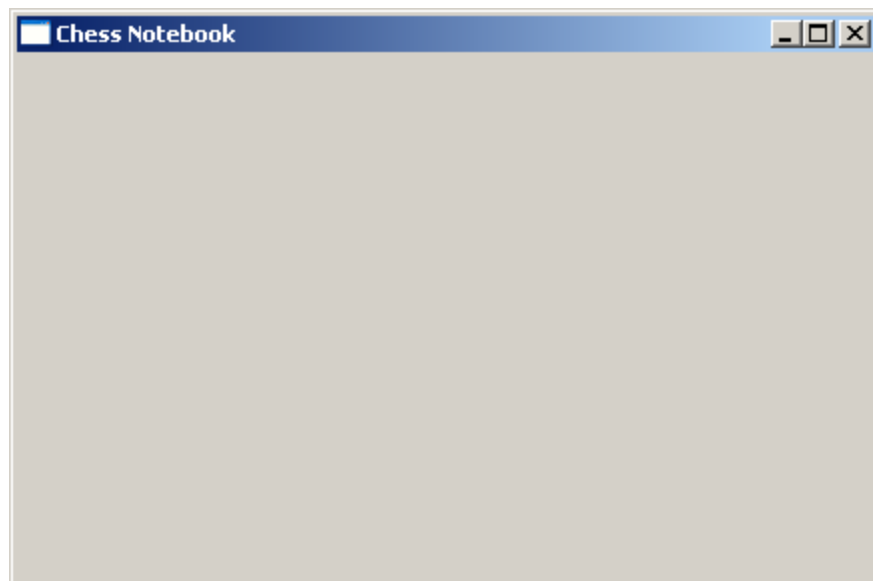


Шахматы. «Минимальные» шахматы на экране

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- Нажать “запуск в отладочном режиме” **F11**



Пиктограмма приложения

```
import org.eclipse.swt.widgets.*;

public class Chess {
    public static void main(String[] args) {
        final Display display = new Display();
        ChessImages.load(display);
        final Shell shell = new Shell(display);
        shell.setSize(600, 500);
        shell.setText("Chess Notebook");
        shell.setImage(ChessImages.iconChessNotebook);
        shell.open();
        while (!shell.isDisposed()) {
            if (!display.readAndDispatch())
                display.sleep();
        }
        display.dispose();
    } // main
} // class Chess
```

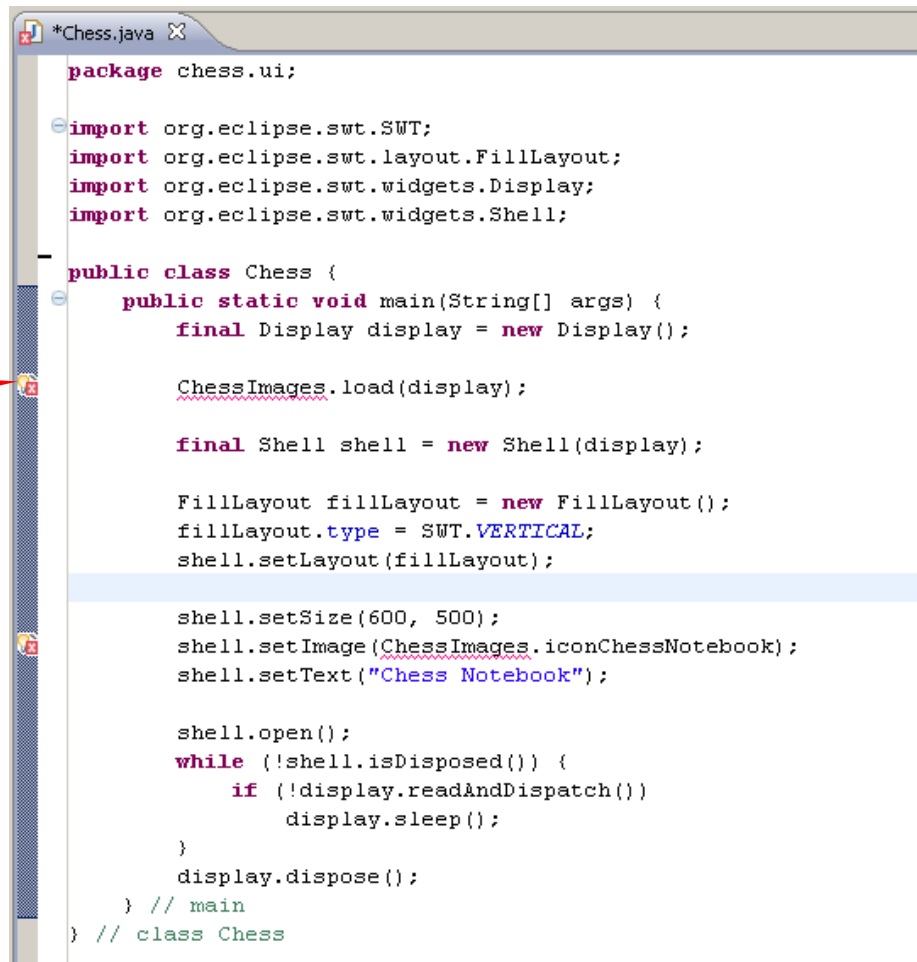
Шахматы. Создание класса *ChessImages*

в режиме «исправления ошибок»

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Нажатие мышью
или
CTRL+1



```
*Chess.java
package chess.ui;

import org.eclipse.swt.SWT;
import org.eclipse.swt.layout.FillLayout;
import org.eclipse.swt.widgets.Display;
import org.eclipse.swt.widgets.Shell;

public class Chess {
    public static void main(String[] args) {
        final Display display = new Display();

        ChessImages.load(display);

        final Shell shell = new Shell(display);

        FillLayout fillLayout = new FillLayout();
        fillLayout.type = SWT.VERTICAL;
        shell.setLayout(fillLayout);

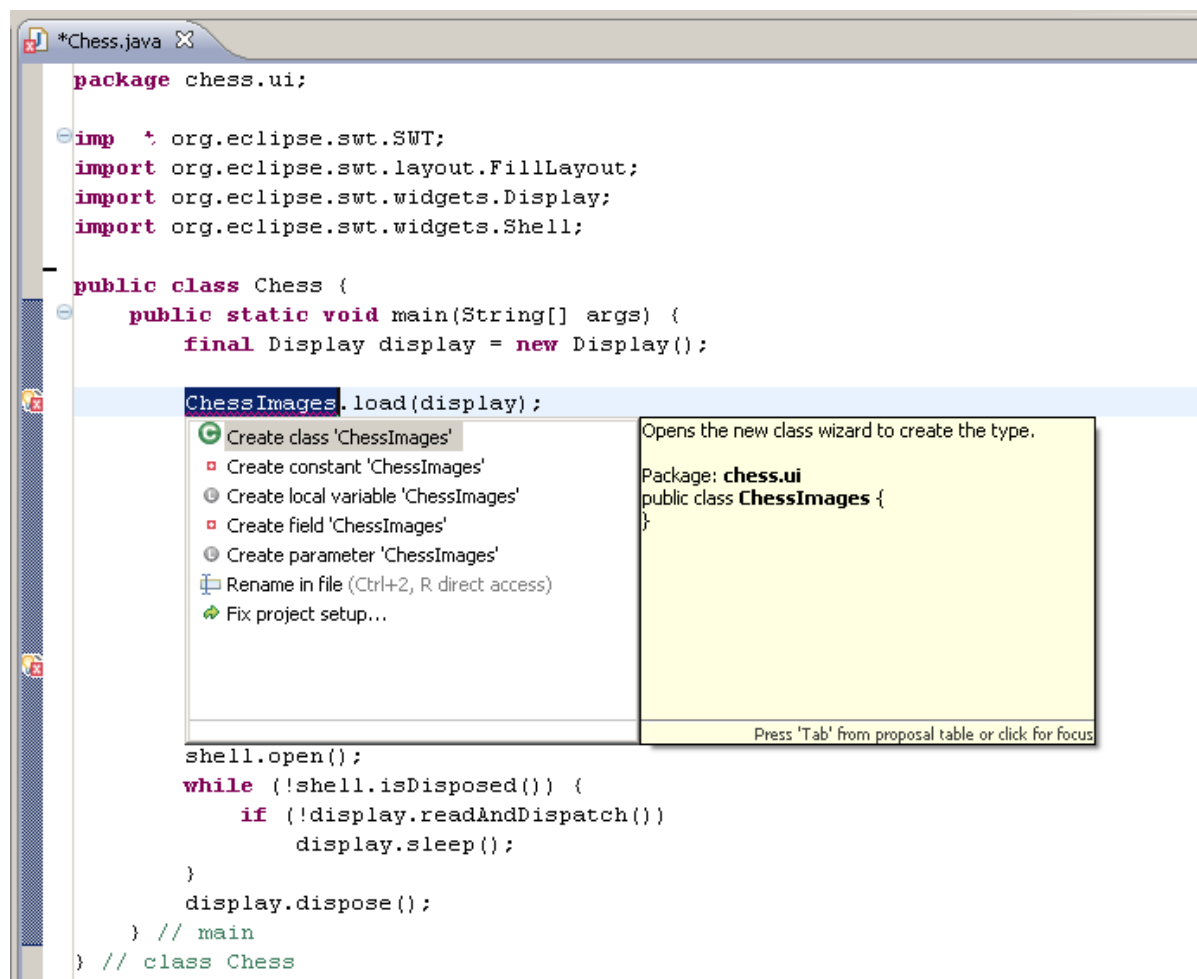
        shell.setSize(600, 500);
        shell.setImage(ChessImages.iconChessNotebook);
        shell.setText("Chess Notebook");

        shell.open();
        while (!shell.isDisposed()) {
            if (!display.readAndDispatch())
                display.sleep();
        }
        display.dispose();
    } // main
} // class Chess
```

Шахматы. Варианты «исправления ошибки» предлагаемые в среде Eclipse

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

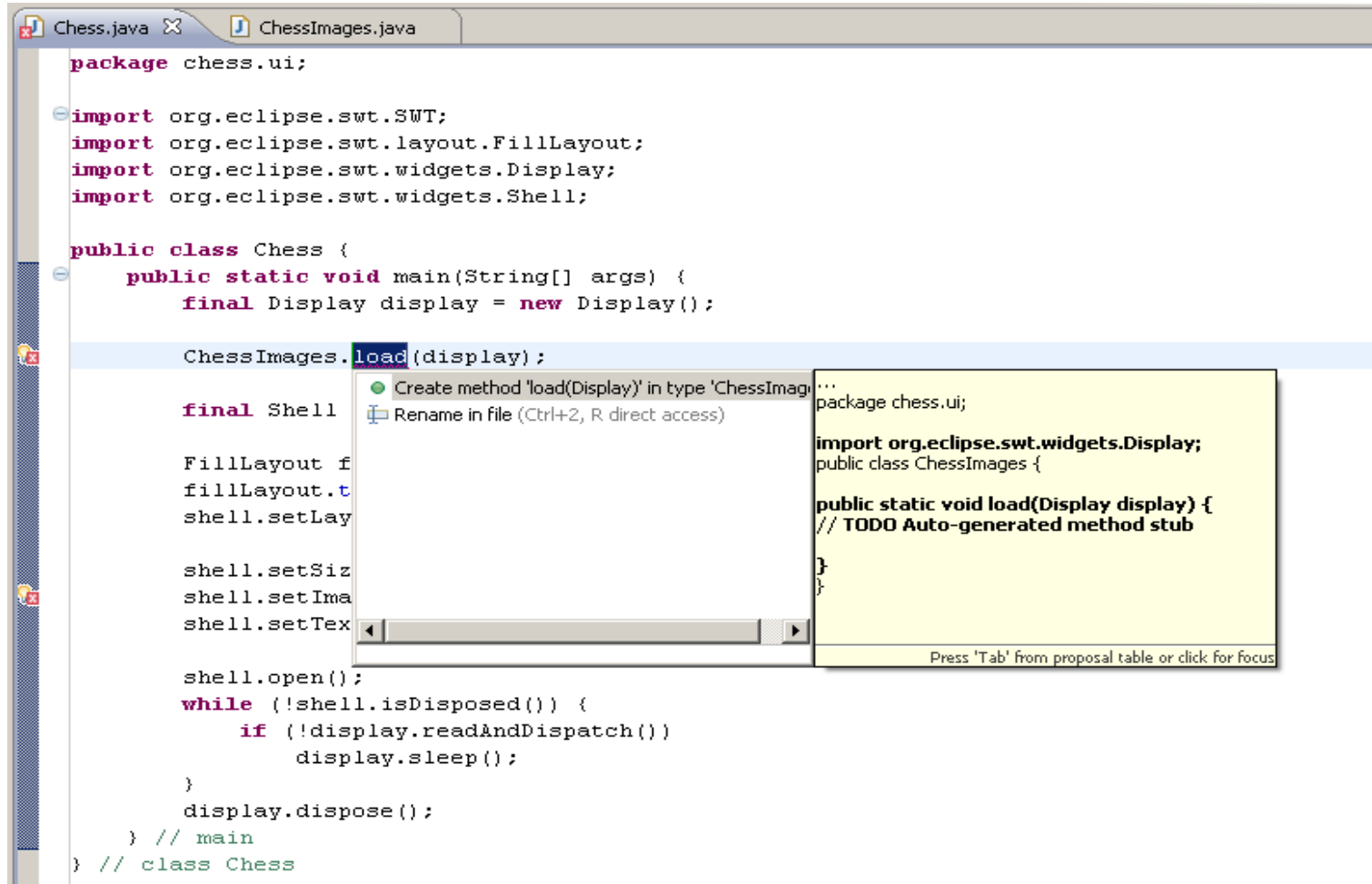


Шахматы. Создание метода *load*

в классе *ChessImages*

МГУ им. М.В.Ломоносова. Факультет ВМК.

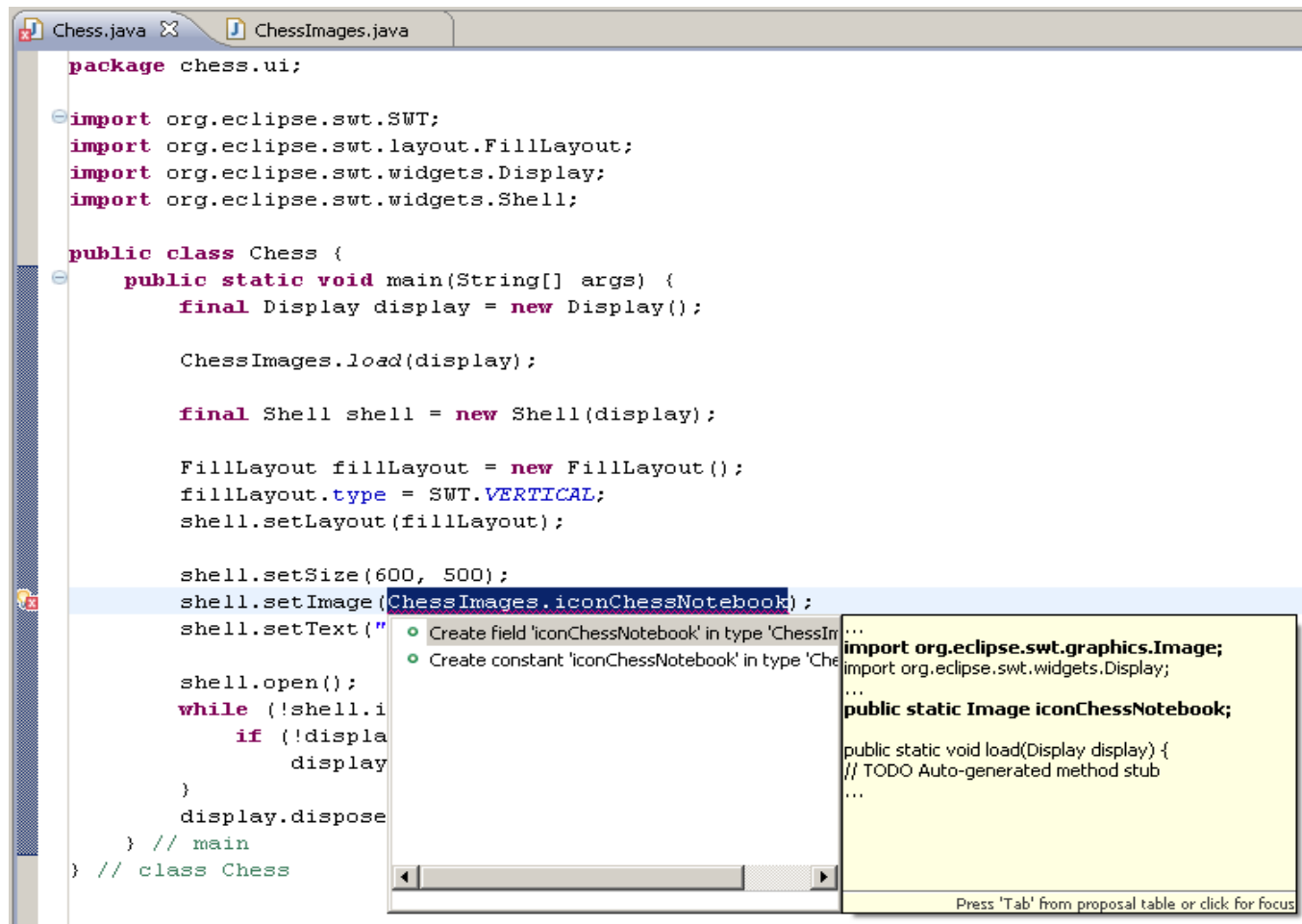
Романов Владимир Юрьевич ©2025



Создание поля *iconChessNotebook* в классе *ChessImages*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Сгенерированный класс *ChessImages*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
package chess.ui;

import org.eclipse.swt.graphics.Image;
import org.eclipse.swt.widgets.Display;

public class ChessImages {

    public static Image iconChessNotebook;

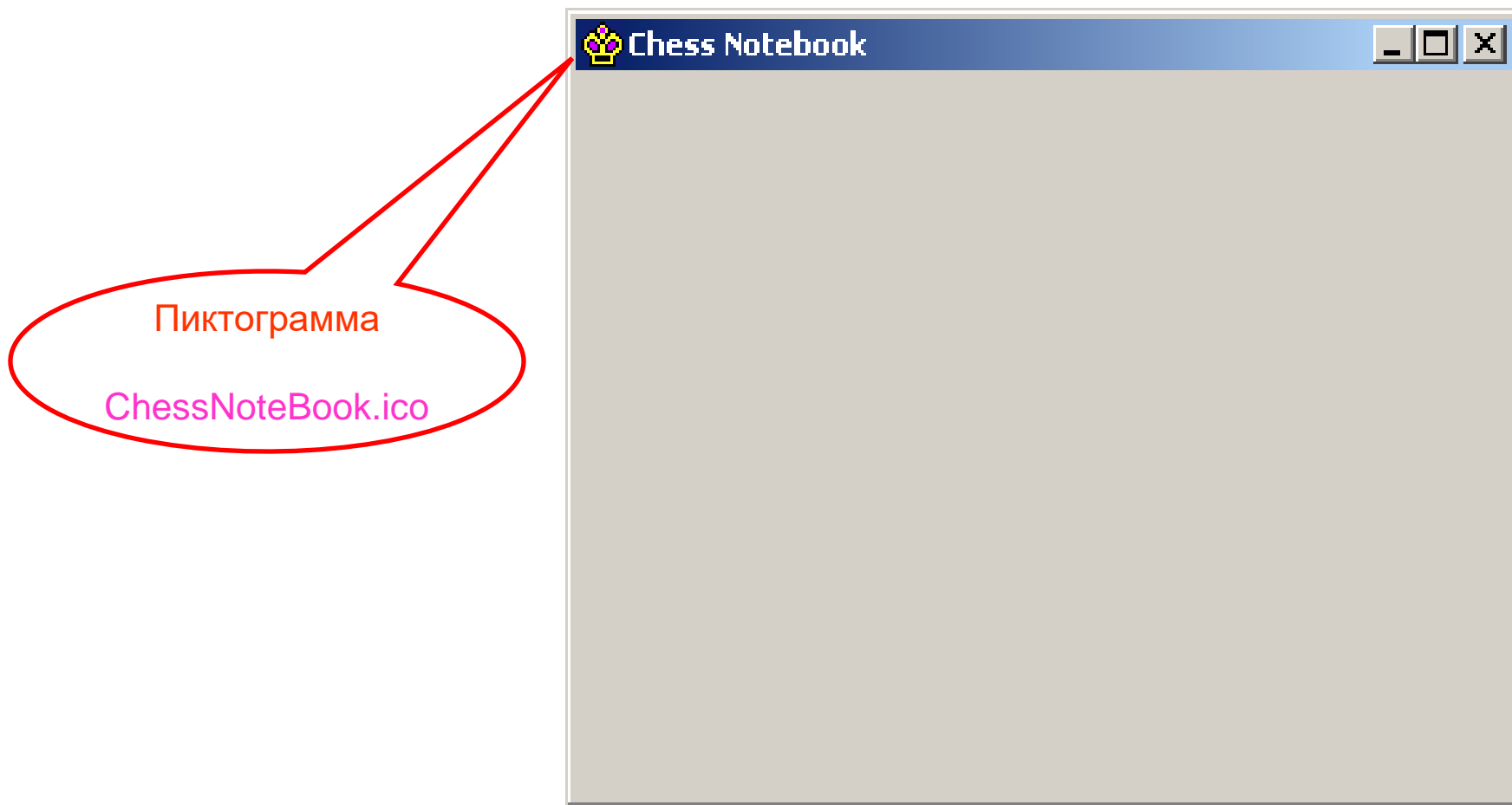
    public static void load(Display display) {
        iconChessNotebook = new Image(display,
            ChessImages.class.getResourceAsStream("ChessNoteBook.ico"));
    }
}
```

- Файл *ChessNoteBook.ico* должен находиться в той же папке, что и файл *ChessImages.class*

Шахматы. Блокнот с пиктограммой

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



SWT. Пакеты библиотеки

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

■ Пакеты библиотеки SWT на языке Java



- ❑ org.eclipse.swt
- ❑ org.eclipse.swt.widgets
- ❑ org.eclipse.swt.graphics
- ❑ org.eclipse.swt.events
- ❑ org.eclipse.swt.layout
- ❑ org.eclipse.swt.dnd
- ❑ org.eclipse.swt.printing
- ❑ org.eclipse.swt.program
- ❑ org.eclipse.swt.accessibility
- ❑ org.eclipse.swt.custom
- ❑ org.eclipse.swt.browser
- ❑ org.eclipse.swt.awt
- ❑ org.eclipse.swt.internal

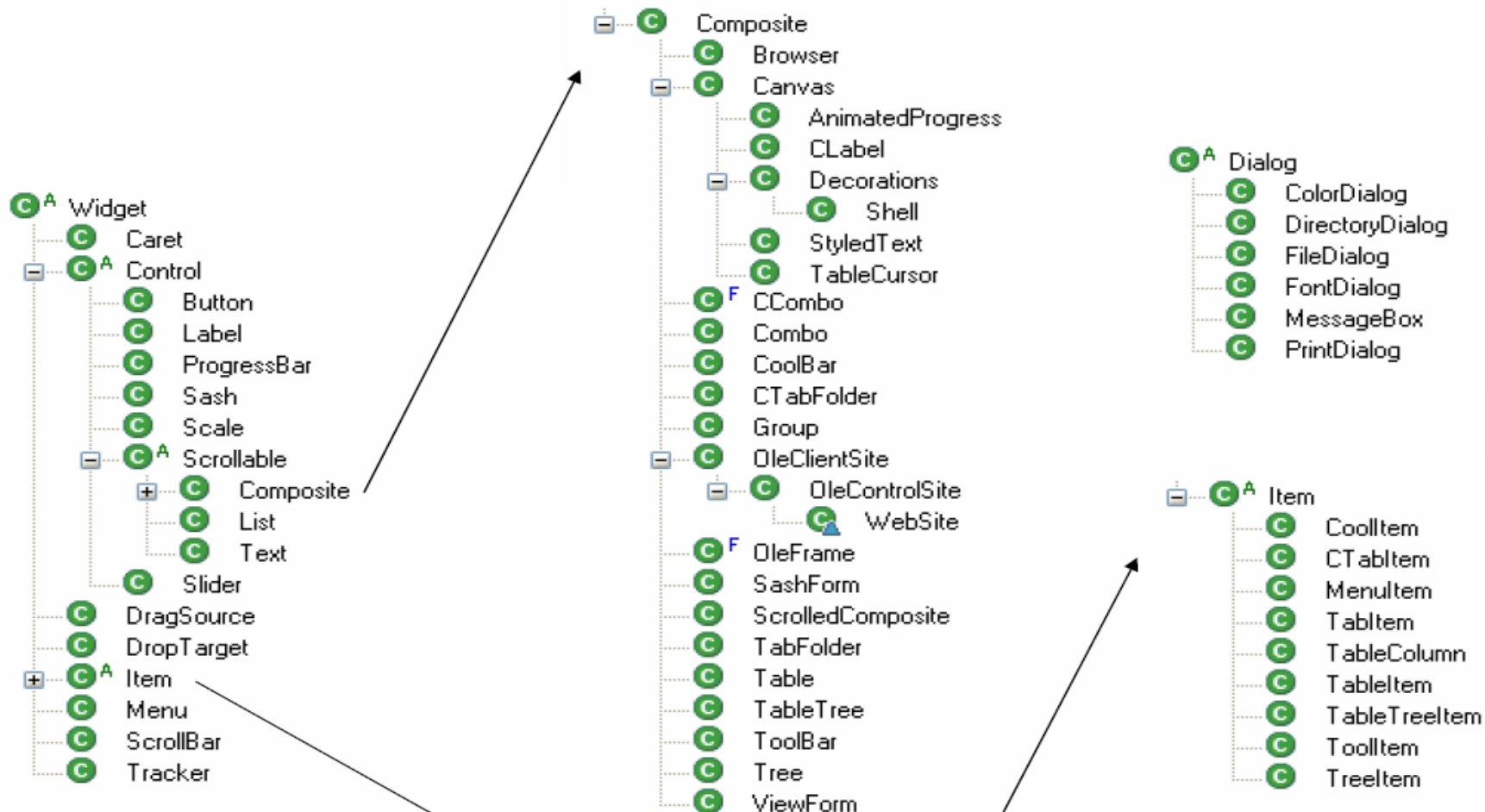
- Содержит все константы библиотеки
 - ❑ `SWT.PUSH, SWT.RADIO`
 - ❑ `SWT.Selection`

- Содержит универсальные методы
 - ❑ `getPlatform()`
 - ❑ `getVersion()`
 - ❑ `error()`

Иерархия управляющих элементов (widgets).

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Конструкторы и стили управляющих элементов.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- *Управляющий элемент (widget)* всегда имеет предка
- Вид типичного конструктора:
`Widget(Composite parent, int style)`
- Стили задаются с помощью констант из класса **SWT**
- Примеры:
 - ❑ `new Label(shell, SWT.NONE);`
 - ❑ `Button push = new Button(shell, SWT.PUSH);`
 - ❑ `Button radio = new Button(parent, SWT.RADIO);`
 - ❑ `Text text = new Text(group, SWT.SINGLE | SWT.BORDER);`
- Исключение. Класс **Shell** всегда имеет предка **Shell** или **Display**
 - ❑ `Shell shell = new Shell(display, SWT.SHELL_TRIM);`
 - ❑ `Shell dialog = new Shell(shell, SWT.DIALOG_TRIM);`

Класс Widget.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- Абстрактный суперкласс для всех элементов интерфейса пользователя
- Создается с помощью конструкторов (без фабрик)
- При создании занимает ресурсы операционной системы
- Ресурсы освобождаются программно с помощью метода `dispose()`
- Уведомляет слушателей когда происходят с этим управляющим элементом происходят события
- Позволяет хранить специфичные для приложения данные
 - `setData(Object)`
 - `setData(String, Object)`
- Событие
 - `Dispose`

Освобождение ресурсов графики и управляющих элементов.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- Вы должны явно освобождать объекты потребляющие ресурсы:
 - Класс **Widget** и его подклассов
 - Классы **Color**, **Cursor**, **Font**, **GC**, **Image**, **Region**,
 - Класс **Device** и его подклассы (**Display**, **Printer**)
- **Правило 1: “Если вы создали его, вы его освобождаете”**
 - Программист должен освободить шрифт:

```
Font font = new Font (display, "Courier", 10, SWT.NORMAL);  
font.dispose ();
```

- Программист не должен освободить шрифт:

```
Font font = control.getFont();
```

Освобождение ресурсов управляющих ЭЛЕМЕНТОВ ИХ ПРЕДКАМИ.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- **Правило 2: “Освобождение предка освобождает его потомков “**
 - ❑ `shell.dispose();` // Освобождает всех потомков окна
 - ❑ `menu.dispose();` // Освобождает все элементы меню
 - ❑ `tree.dispose();` // Освобождает все элементы дерева

- **Заметим, что:**
 - ❑ `control.dispose();`
 - ❑ `menulitem.dispose();`
 - ❑ Освобождает элемент меню созданный с помощью `setMenu(menu);`

Класс Control

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

- Абстрактный суперкласс для всех легковесных (heavyweight) элементов интерфейса пользователя
- Стили
 - BORDER, LEFT_TO_RIGHT, RIGHT_TO_LEFT
- События
 - FocusIn, FocusOut
 - KeyDown, KeyUp
 - Traverse
 - MouseDown, MouseUp, MouseDoubleClick
 - MouseEnter, MouseExit, MouseMove, MouseHover
 - Move, Resize
 - Paint
 - Help

Класс Shell

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

■ Конструкторы

- **new** Shell(display, SWT.SHELL_TRIM);
- **new** Shell(shell, SWT.DIALOG_TRIM);

■ Стили

- BORDER, CLOSE, MIN, MAX, NO_TRIM, RESIZE, TITLE
- APPLICATION_MODAL, MODELESS, PRIMARY_MODAL, SYSTEM_MODAL

■ События

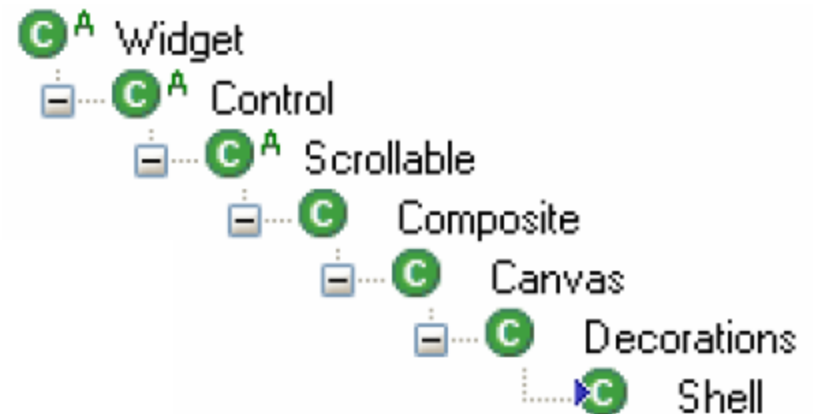
- Close, Activate, Deactivate, Iconify, Deiconify

■ Характерные методы

- open(), close(), setActive()

■ Замечания

- Предок для *shell* верхнего уровня всегда Display
- Предок для *shell* - диалогового окна всегда *Shell* верхнего уровня



Класс Composite

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

■ Конструкторы

- `new Composite(parent, SWT.NONE);`

■ Стили

- `NO_BACKGROUND, NO_FOCUS, NO_MERGE_PAINTS, NO_REDRAW_RESIZE, NO_RADIO_GROUP`

■ События

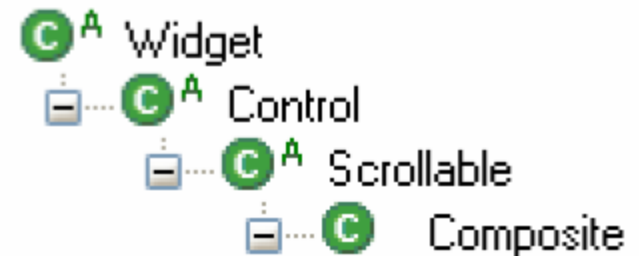
- `Close, Activate, Deactivate, Iconify, Deiconify`

■ Характерные методы

- `getChildren()`
- `setLayout(Layout), layout(boolean)`
- `setTabList(Control[])`

■ Замечания

- может иметь потомков – управляющие элементы (*controls*)
- может использовать класс `Layout` для задания положения потомков
- используется как суперкласс для создания управляющих элементов (*widgets*) пользователя



Класс Canvas

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

■ Конструкторы

- ❑ `new Canvas(parent, SWT.NONE);`

■ Стили

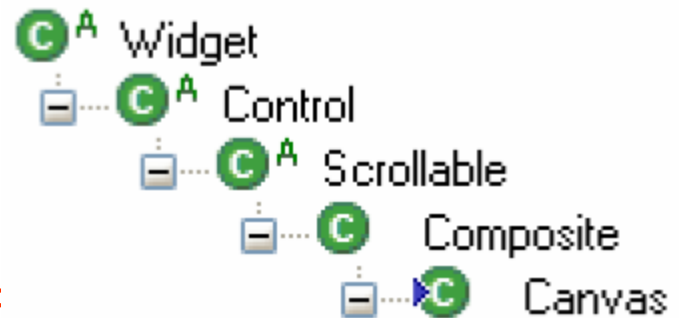
- ❑ `NO_BACKGROUND, NO_FOCUS, NO_MERGE_PAINTS, NO_REDRAW_RESIZE`

■ Характерные методы

- ❑ `scroll(int, int, int, int, int, int, boolean)`
- ❑ `setCaret(Caret)`

■ Замечания

- ❑ Обычно используется как «чистый лист бумаги» для рисования графики
- ❑ используется как суперкласс для создания управляющих элементов (*widgets*) пользователя



Класс TabFolder

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

■ Конструкторы

- **new** TabFolder(parent, SWT.TOP);

■ Стили

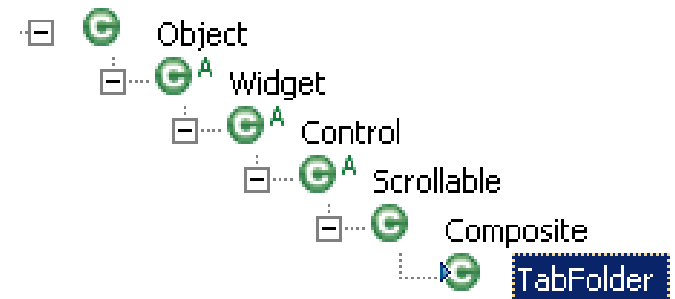
- **SWT.TOP** , **SWT.BOTTOM**, **SWT.LEFT**, **SWT.RIGHT**

■ Характерные методы

- **setSelection(int, boolean)**
- **setSelection(TabItem)**
- **TabItem getSelection()**
- **int getSelectionIndex()**

■ Замечания

- используется как «книга» с закладками



Класс TabItem

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

■ Конструкторы

- **new** TabItem(parent, SWT.NONE);

■ Стили

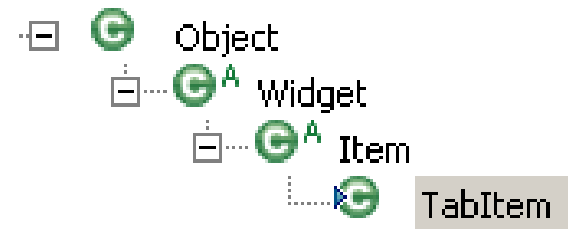
- Унаследованные от родителей

■ Характерные методы

- setControl(Control)
- setImage(Image)
- setText(String)
- setToolTipText(String)

■ Замечания

- используется как закладка в «книге»

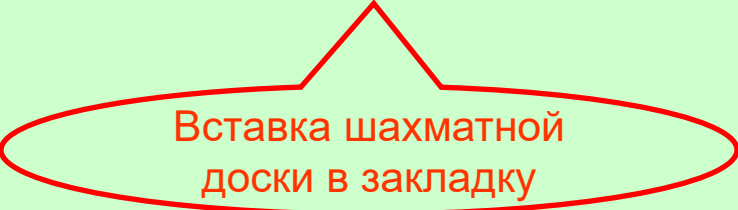


Шахматы. Шахматная доска – теперь как закладка в «книге» игр

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public static void main(String[] args) {  
    final Display display = new Display();  
    ChessImages.load(display);  
  
    final Shell shell = new Shell(display);  
    shell.setSize(600, 500);  
    shell.setText("Chess Notebook");  
    shell.setImage(ChessImages.iconChessNotebook);  
    setLayout(new FillLayout()); // Чтобы растянуть gamesFolder на все окно.  
  
    final TabFolder gamesFolder = new TabFolder(shell, SWT.TOP);  
    TabItem chessItem = new TabItem(gamesFolder, SWT.NULL);  
    chessItem.setText("Шахматы");  
    chessItem.setControl( new ChessBoard(gamesFolder, SWT.NONE) );  
  
    // ...  
} // main
```



Вставка шахматной
доски в закладку

Шахматы. Шахматы – пиктограмма для закладки в блокноте игр

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public static void main(String[] args) {
```

```
// ...
```

```
final TabFolder gamesFolder = new TabFolder(shell, SWT.TOP);
```

```
final Image chessTabImage = new Image(display,  
    ChessImages.imageKnightBlack.getImageData().scaledTo(20, 20) );
```

```
TabItem chessItem = new TabItem(gamesFolder, SWT.NULL);
```

```
chessItem.setText("Шахматы");
```

```
chessItem.setImage(chessTabImage);
```

```
chessItem.setControl( new ChessBoard(gamesFolder, SWT.NONE) );
```

```
// ...
```

```
} // main
```

Масштабирование
рисунка

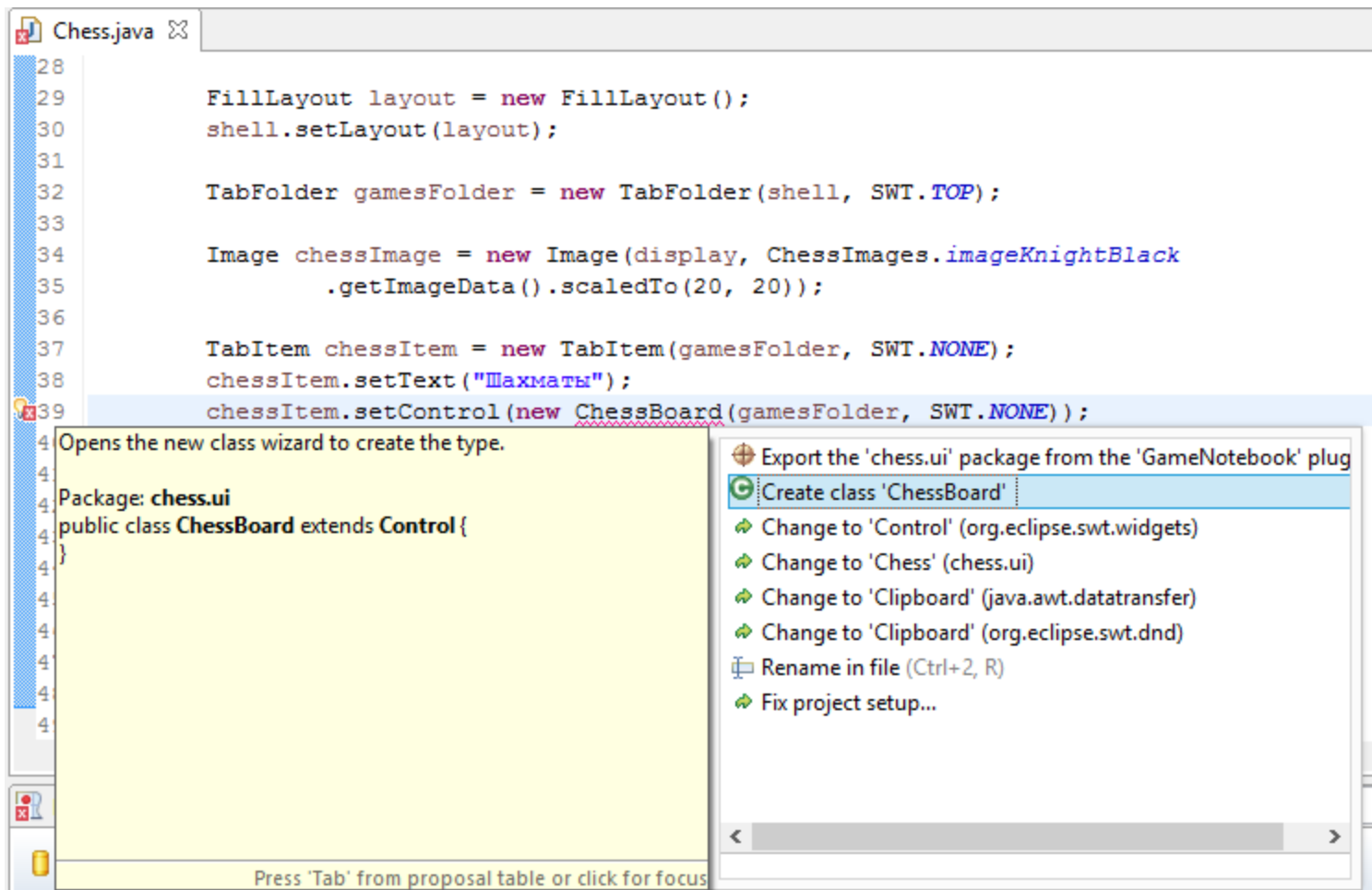
Пиктограмма для
закладки шахмат в
блокноте клеточных игр

Шахматы. Создание класса для шахматной доски с помощью wizard

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Ctrl + 1 когда «каретка» в тексте *ChessBoard*

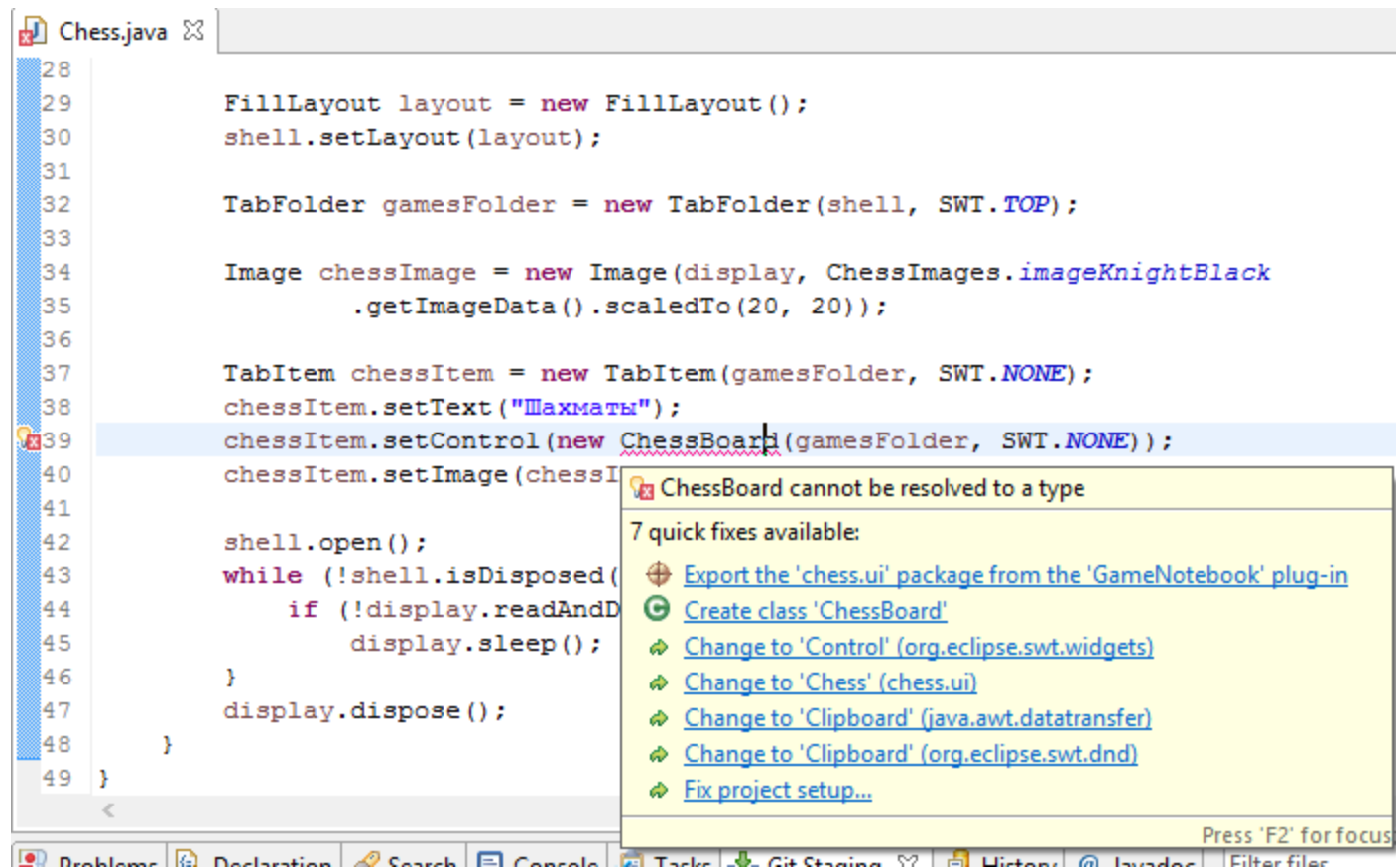


Шахматы. Создание класса для шахматной доски с помощью wizard

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

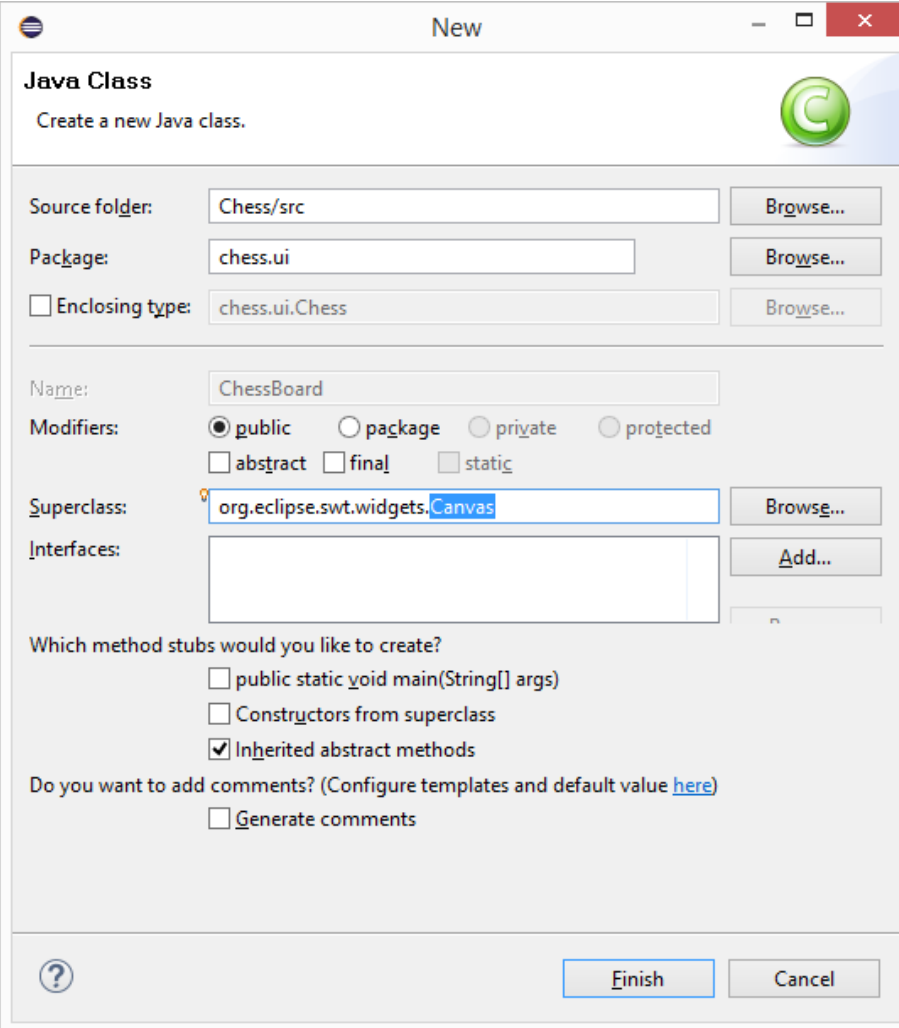
Всплывающая подсказка когда мышка над текстом *ChessBoard*



Шахматы. Создание класса для шахматной доски с помощью wizard

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



The screenshot shows the 'New Java Class' wizard in the Eclipse IDE. The window title is 'New'. The wizard is titled 'Java Class' and has a subtitle 'Create a new Java class.' with a green 'C' icon.

Fields and options:

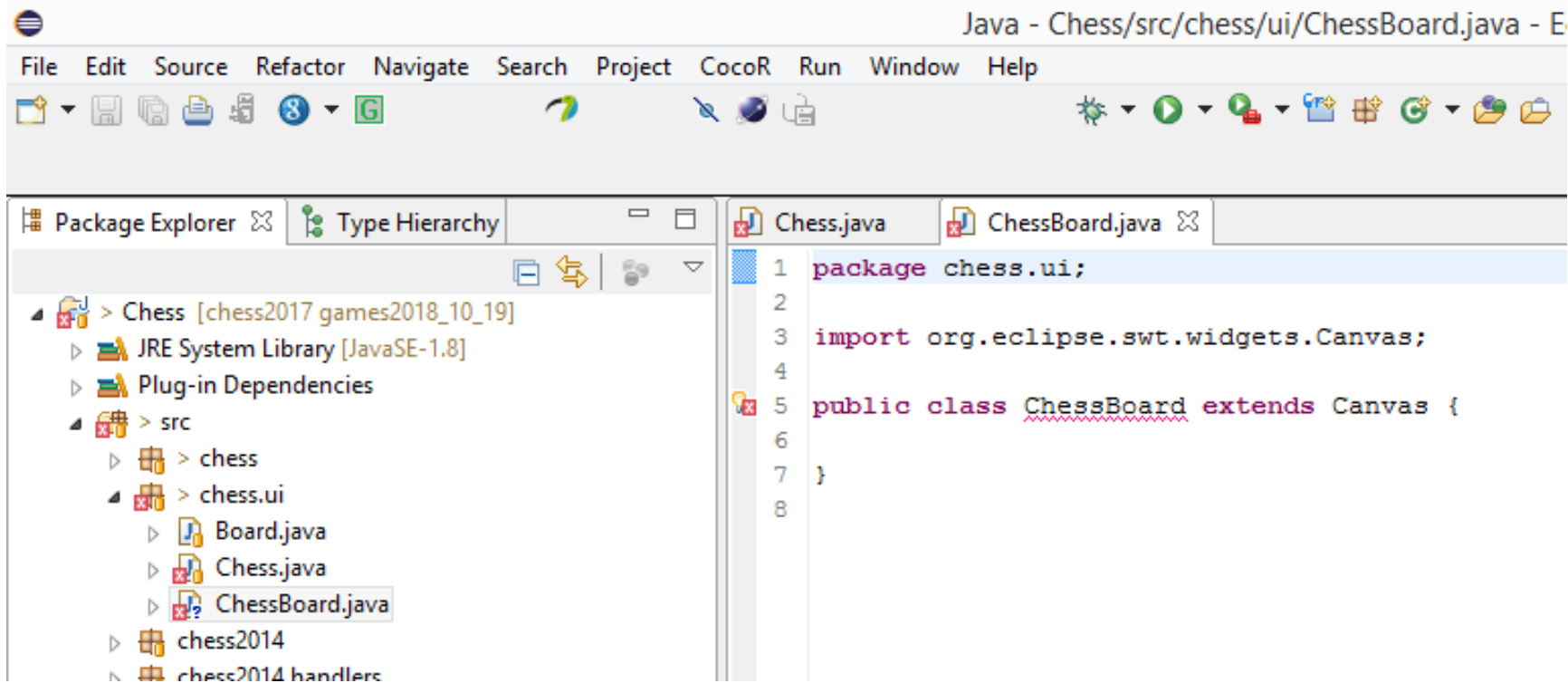
- Source folder:** Chess/src (with a 'Browse...' button)
- Package:** chess.ui (with a 'Browse...' button)
- Enclosing type:** ☐ chess.ui.Chess (with a 'Browse...' button)
- Name:** ChessBoard
- Modifiers:** ☒ public, ☐ package, ☐ private, ☐ protected, ☐ abstract, ☐ final, ☐ static
- Superclass:** org.eclipse.swt.widgets.Canvas (with a 'Browse...' button)
- Interfaces:** (empty list with an 'Add...' button)
- Which method stubs would you like to create?**
 - ☐ public static void main(String[] args)
 - ☐ Constructors from superclass
 - ☒ Inherited abstract methods
- Do you want to add comments?** (Configure templates and default value [here](#))
 - ☐ Generate comments

At the bottom, there is a help icon (?), a 'Finish' button, and a 'Cancel' button.

Шахматы. Класс *ChessBoard* создан

МГУ им. М.В.Ломоносова. Факультет ВМК.

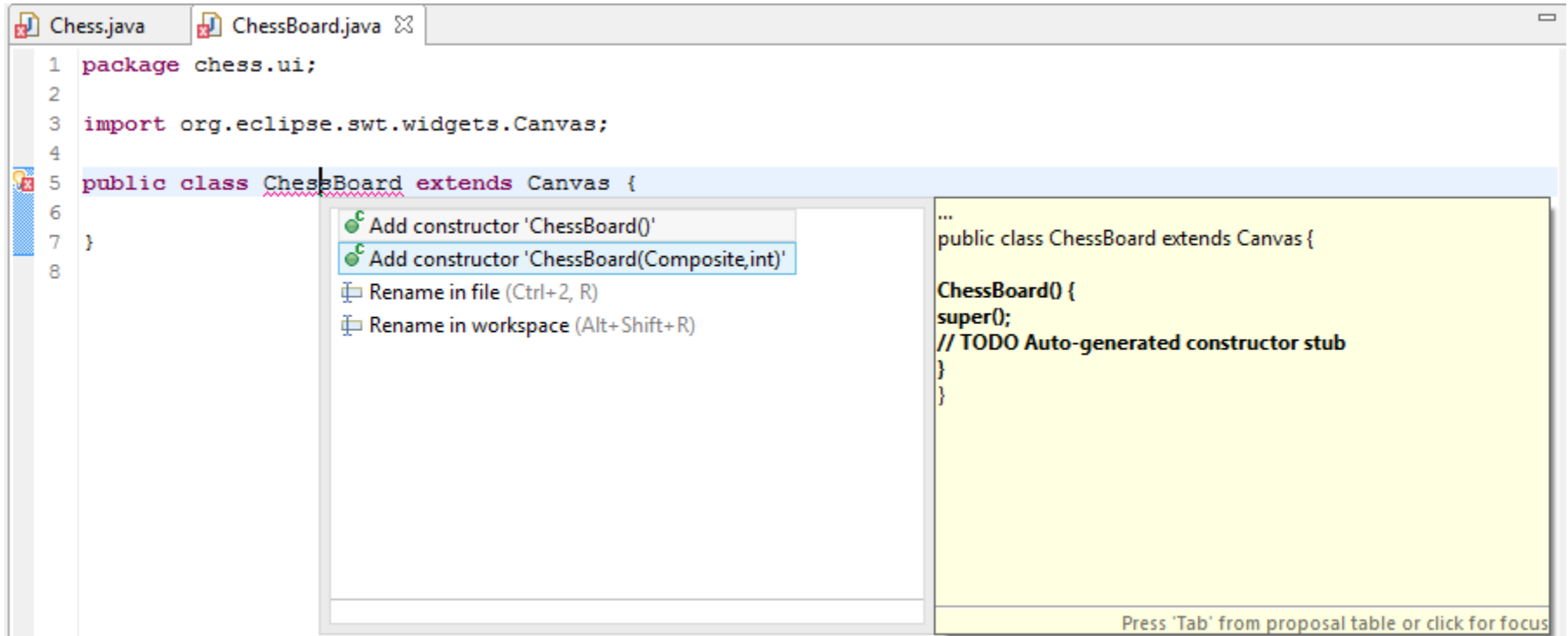
Романов Владимир Юрьевич ©2025



Шахматы. Создание конструктора класса *ChessBoard* с помощью wizard

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



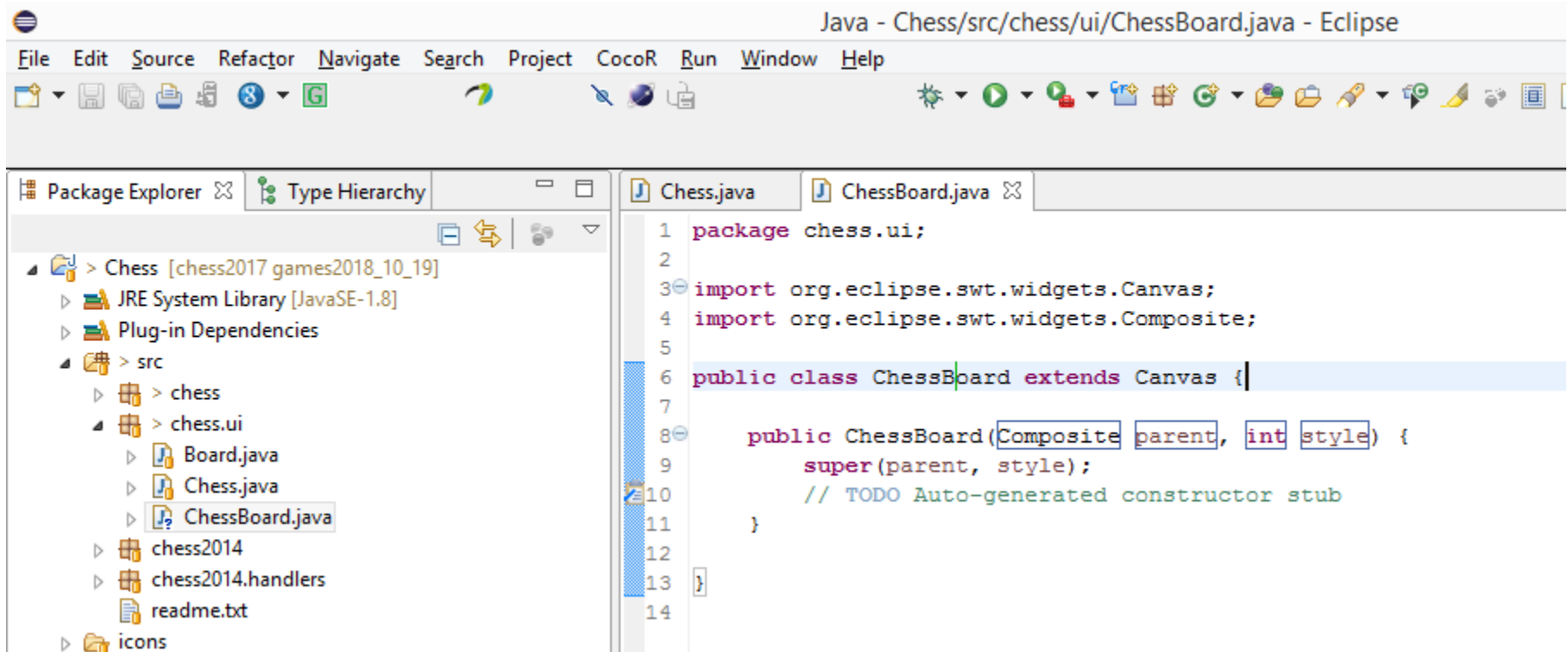
Ctrl + 1 когда «каретка» в тексте *ChessBoard*

Шахматы. Конструктор класса *ChessBoard*

СОЗДАН

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Java - Chess/src/chess/ui/ChessBoard.java - Eclipse

File Edit Source Refactor Navigate Search Project CocoR Run Window Help

Package Explorer Type Hierarchy

Chess [chess2017 games2018_10_19]

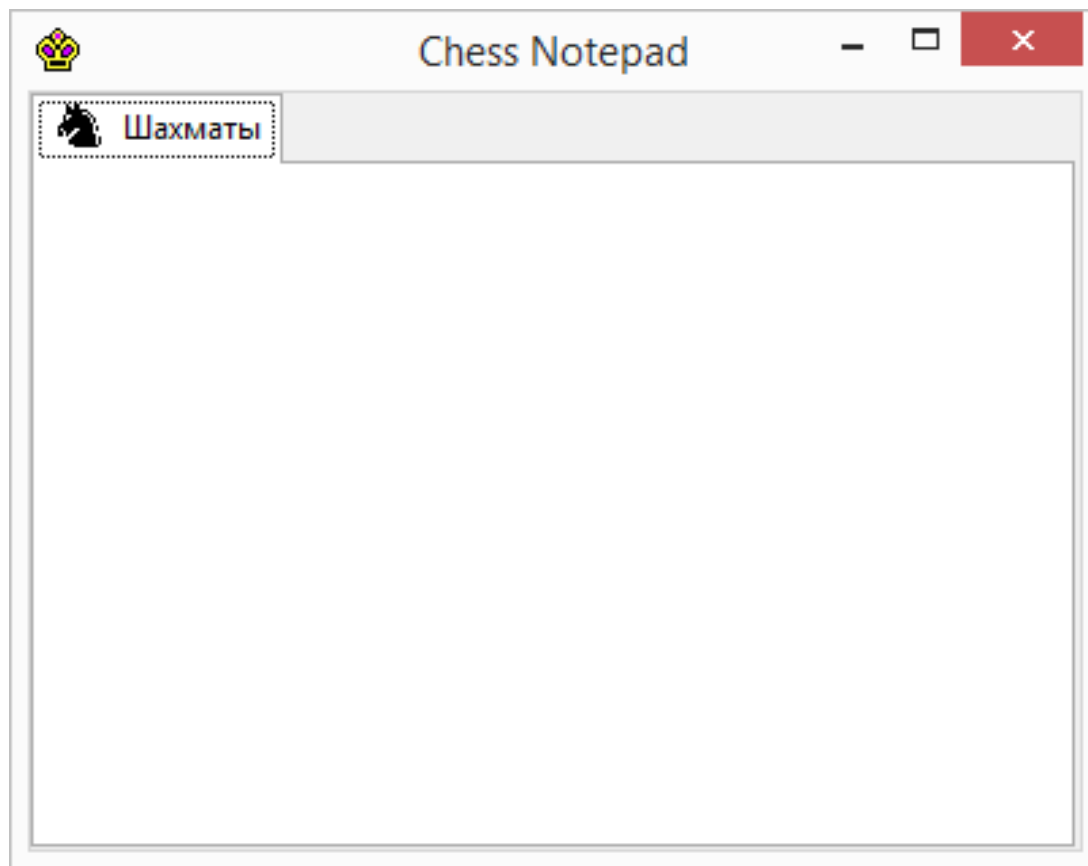
- JRE System Library [JavaSE-1.8]
- Plug-in Dependencies
- src
 - chess
 - chess.ui
 - Board.java
 - Chess.java
 - ChessBoard.java
 - chess2014
 - chess2014.handlers
 - readme.txt
 - icons

```
1 package chess.ui;
2
3 import org.eclipse.swt.widgets.Canvas;
4 import org.eclipse.swt.widgets.Composite;
5
6 public class ChessBoard extends Canvas {
7
8     public ChessBoard(Composite parent, int style) {
9         super(parent, style);
10        // TODO Auto-generated constructor stub
11    }
12
13 }
14
```


Шахматы. Показывается только пустое поле для рисования доски

МГУ им. М.В.Ломоносова. Факультет ВМК.

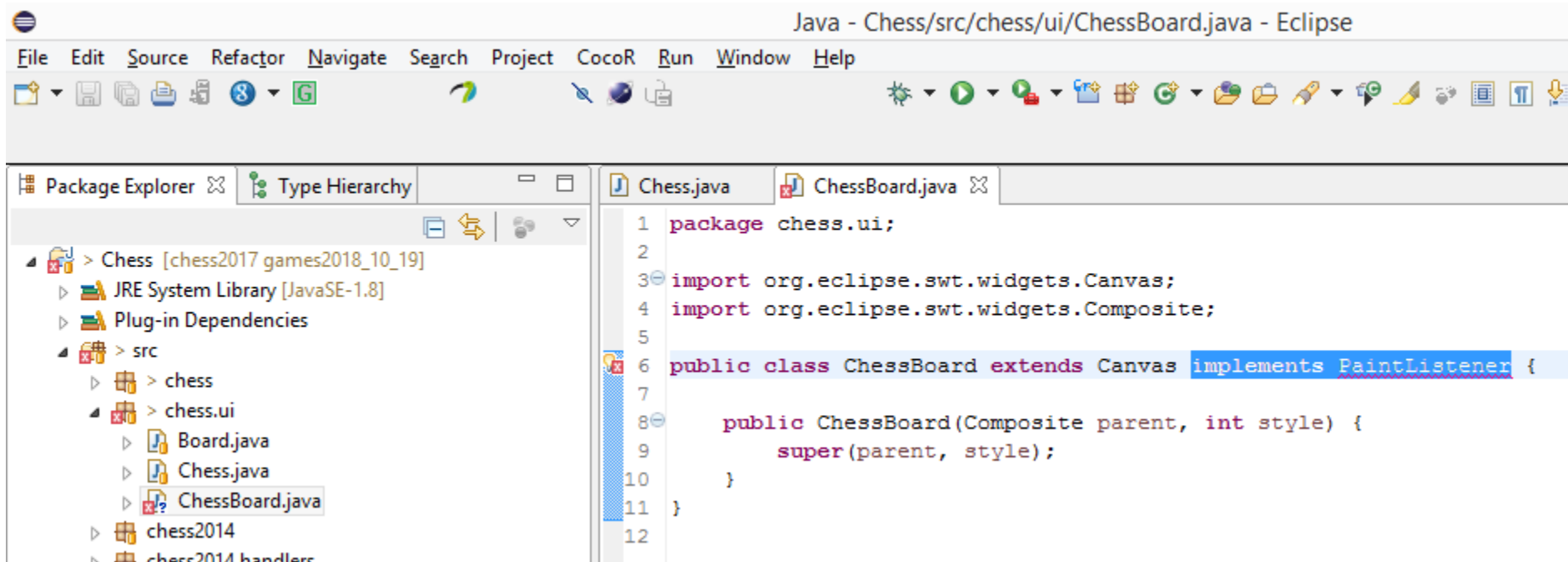
Романов Владимир Юрьевич ©2025



Шахматы. Рисование рамки доски

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



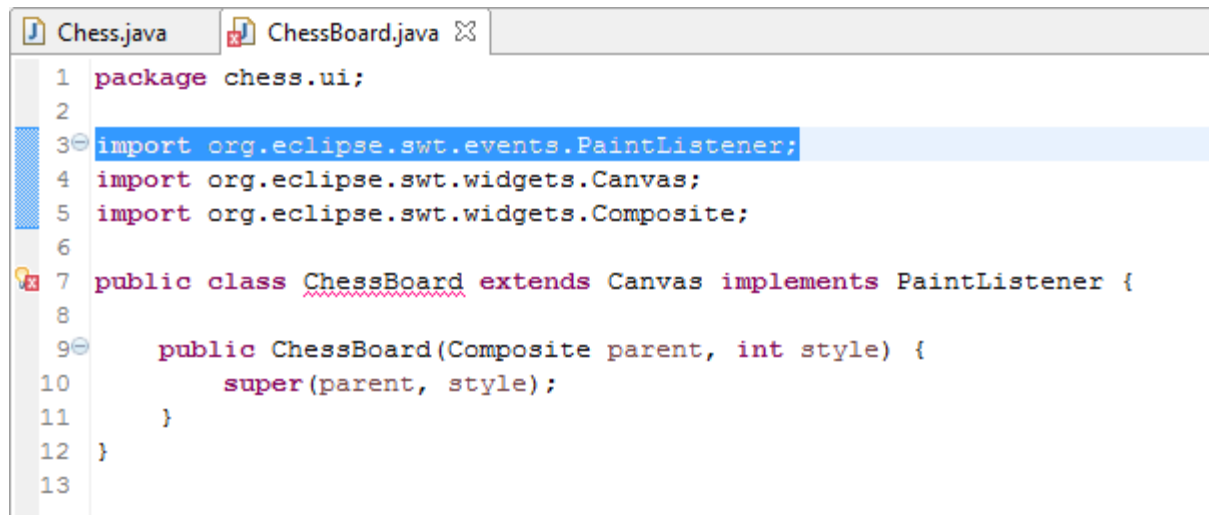
The screenshot shows the Eclipse IDE interface. The title bar reads "Java - Chess/src/chess/ui/ChessBoard.java - Eclipse". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, CocoR, Run, Window, and Help. The toolbar contains various icons for file operations and development tools. The Package Explorer on the left shows the project structure: Chess [chess2017 games2018_10_19] > JRE System Library [JavaSE-1.8] > Plug-in Dependencies > src > chess > chess.ui > ChessBoard.java. The main editor window displays the code for ChessBoard.java, which is a Java class extending Canvas and implementing PaintListener. The code is as follows:

```
1 package chess.ui;
2
3 import org.eclipse.swt.widgets.Canvas;
4 import org.eclipse.swt.widgets.Composite;
5
6 public class ChessBoard extends Canvas implements PaintListener {
7
8     public ChessBoard(Composite parent, int style) {
9         super(parent, style);
10    }
11 }
12
```

Шахматы. Вставка импорта неизвестного класса или интерфейса

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



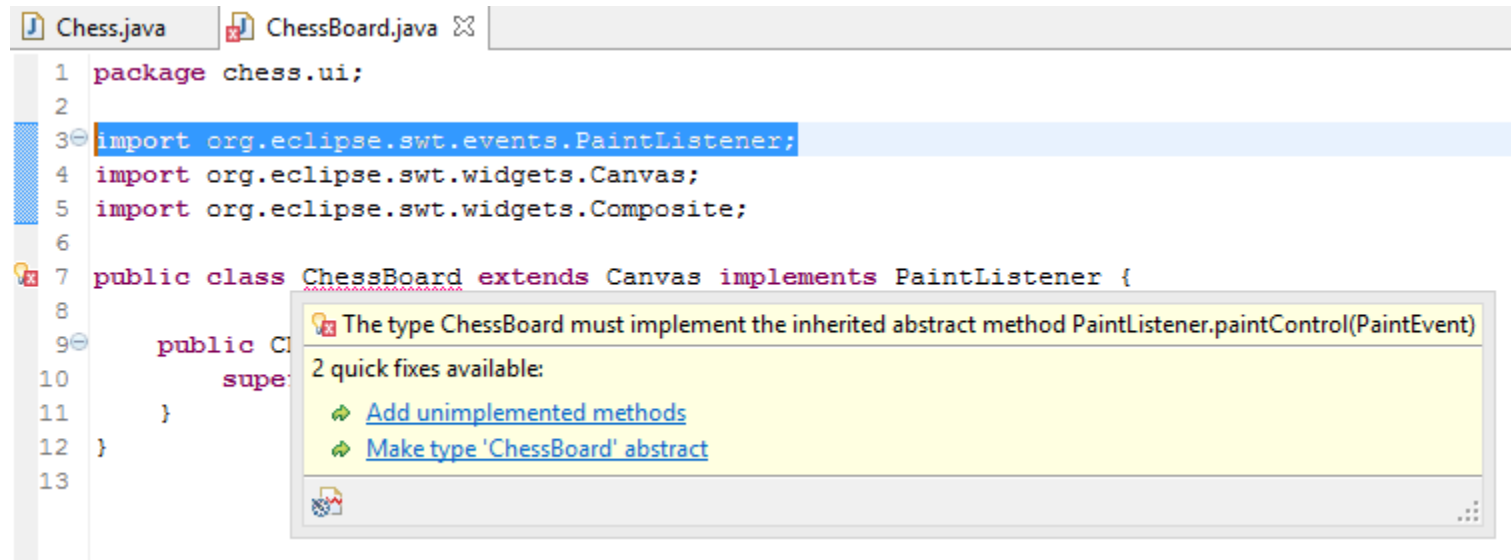
```
Chess.java ChessBoard.java ✕
1 package chess.ui;
2
3 import org.eclipse.swt.events.PaintListener;
4 import org.eclipse.swt.widgets.Canvas;
5 import org.eclipse.swt.widgets.Composite;
6
7 public class ChessBoard extends Canvas implements PaintListener {
8
9     public ChessBoard(Composite parent, int style) {
10         super(parent, style);
11     }
12 }
13
```

Ctrl + Shift + O для вставки импорта с помощью wizard

Шахматы. Класс *ChessBoard* реализует интерфейс *PaintListener*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



The screenshot shows an IDE window with two tabs: `Chess.java` and `ChessBoard.java`. The `ChessBoard.java` tab is active, displaying the following code:

```
1 package chess.ui;
2
3 import org.eclipse.swt.events.PaintListener;
4 import org.eclipse.swt.widgets.Canvas;
5 import org.eclipse.swt.widgets.Composite;
6
7 public class ChessBoard extends Canvas implements PaintListener {
8
9     public ChessBoard() {
10         super();
11     }
12 }
13
```

An IDE error message is displayed over the code, stating: "The type ChessBoard must implement the inherited abstract method PaintListener.paintControl(PaintEvent)". Below the message, it offers "2 quick fixes available":

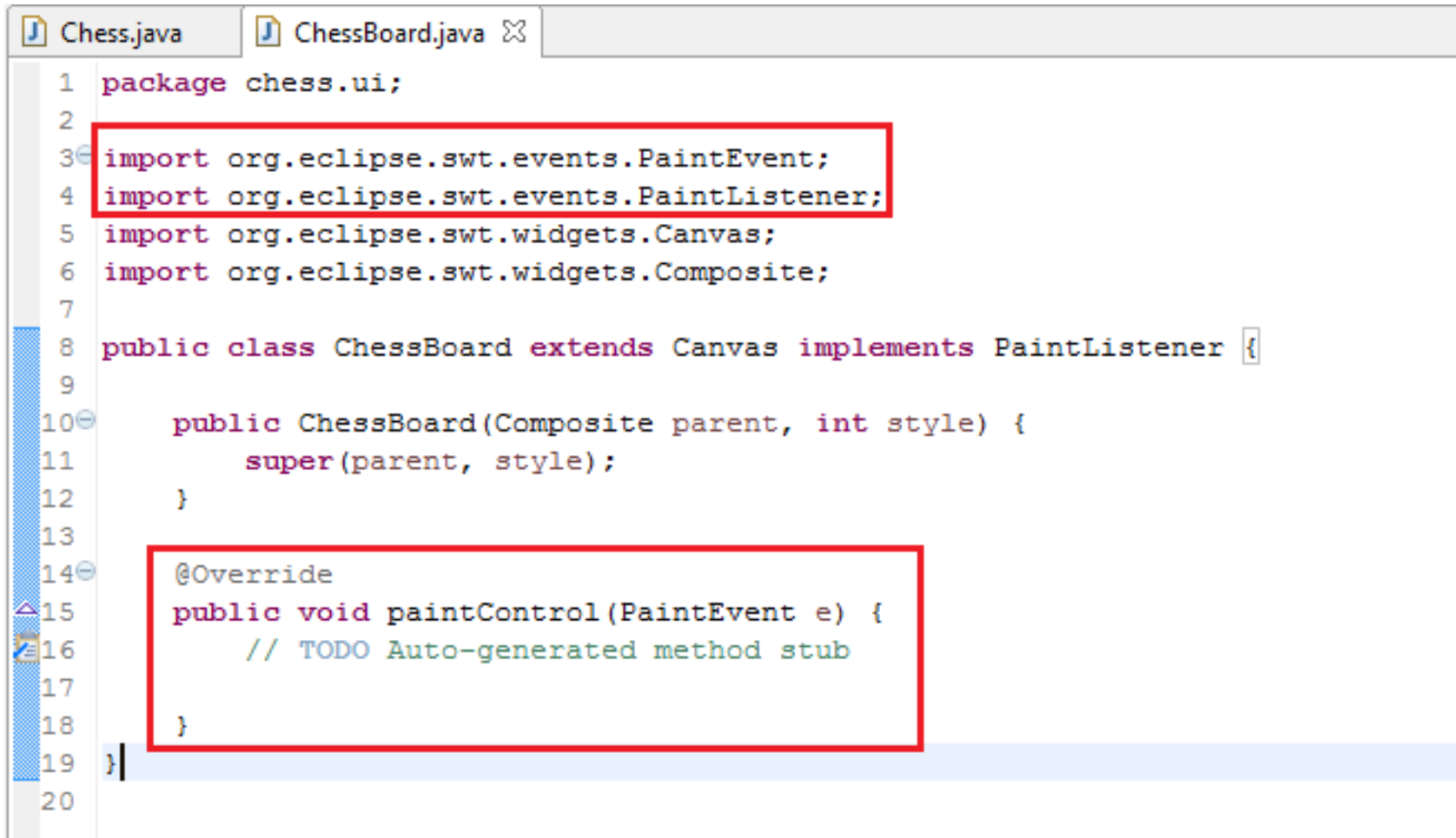
- [Add unimplemented methods](#)
- [Make type 'ChessBoard' abstract](#)

Ctrl + Shift + O добавление нереализованных методов с помощью wizard

Шахматы. Добавлена реализация методов интерфейса *PaintListener*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



```
1 package chess.ui;
2
3 import org.eclipse.swt.events.PaintEvent;
4 import org.eclipse.swt.events.PaintListener;
5 import org.eclipse.swt.widgets.Canvas;
6 import org.eclipse.swt.widgets.Composite;
7
8 public class ChessBoard extends Canvas implements PaintListener {
9
10     public ChessBoard(Composite parent, int style) {
11         super(parent, style);
12     }
13
14     @Override
15     public void paintControl(PaintEvent e) {
16         // TODO Auto-generated method stub
17     }
18
19 }
20
```

Шахматы. Рисование рамки по границам ДОСКИ

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public void paintControl(PaintEvent e) {  
    Rectangle clientArea = getClientArea();  
    e.gc.drawRectangle(0, 0, clientArea.width - 1, clientArea.height - 1);  
}
```

Шахматы. Рисование рамки по границам доски слушателем события *Paint*

МГУ им. М.В.Ломоносова. Факультет ВМК.

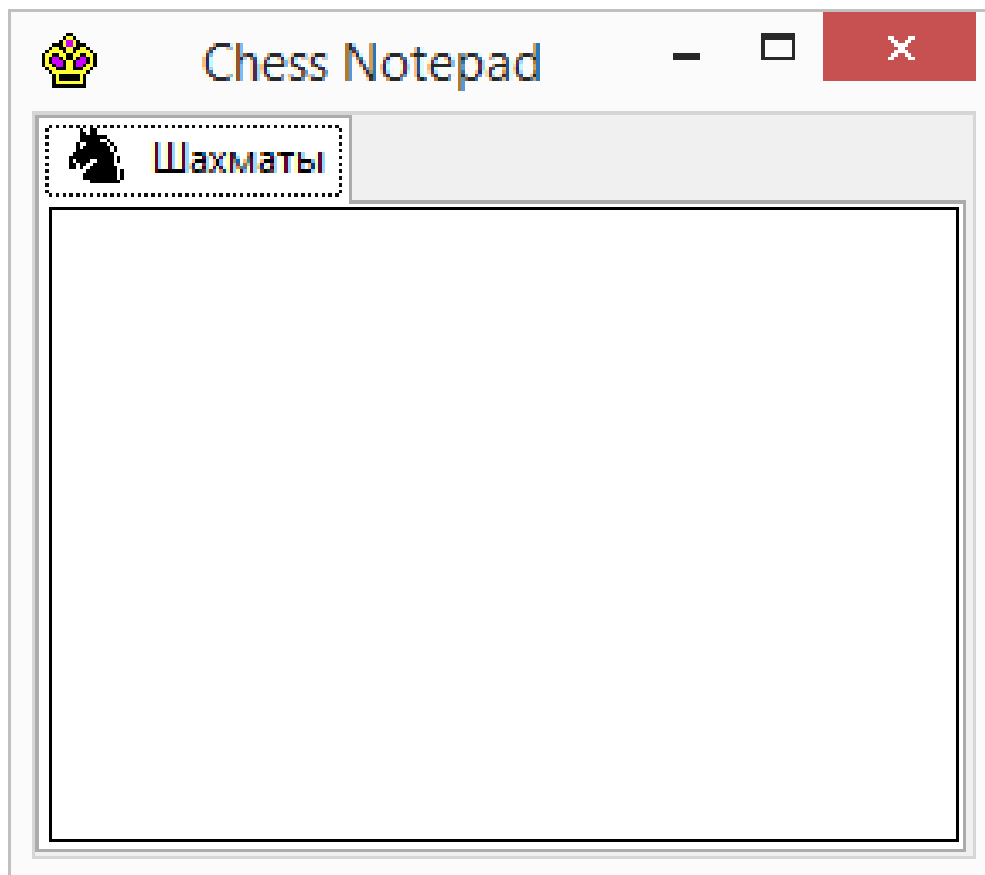
Романов Владимир Юрьевич ©2025

```
Chess.java ChessBoard.java ✕
1 package chess.ui;
2
3 import org.eclipse.swt.events.PaintEvent;
4 import org.eclipse.swt.events.PaintListener;
5 import org.eclipse.swt.graphics.Rectangle;
6 import org.eclipse.swt.widgets.Canvas;
7 import org.eclipse.swt.widgets.Composite;
8
9 public class ChessBoard extends Canvas implements PaintListener {
10
11     public ChessBoard(Composite parent, int style) {
12         super(parent, style);
13         addPaintListener(this);
14     }
15
16     @Override
17     public void paintControl(PaintEvent e) {
18         Rectangle clientArea = getClientArea();
19         e.gc.drawRectangle(0, 0, clientArea.width - 1, clientArea.height - 1);
20     }
21 }
22
```

Шахматы. Рамка доски

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



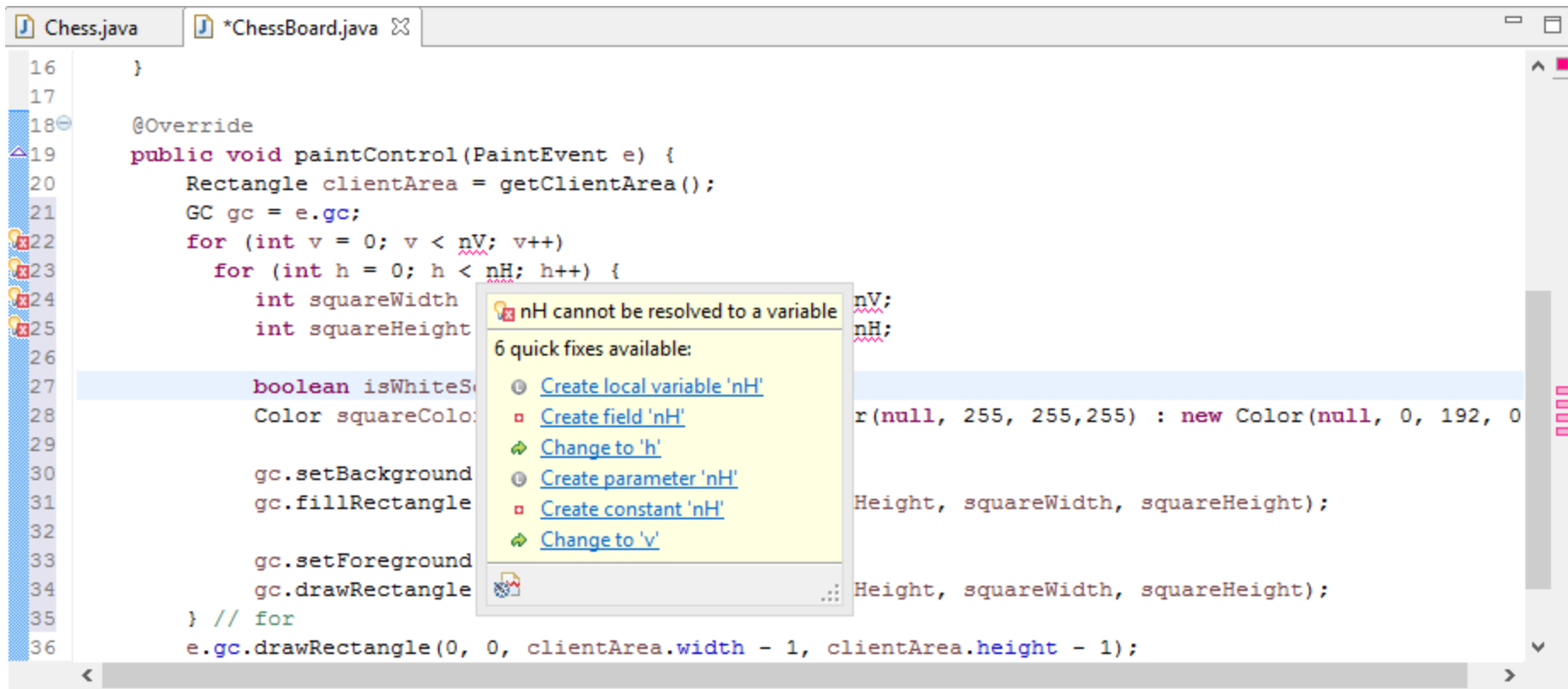
Шахматы. Отрисовка клеток доски

```
public void paintControl(PaintEvent e) {  
    Rectangle clientArea = getClientArea();  
    GC gc = e.gc;  
    for (int v = 0; v < nV; v++)  
        for (int h = 0; h < nH; h++) {  
            int squareWidth = getClientArea().width / nV;  
            int squareHeight = getClientArea().height / nH;  
  
            boolean isWhiteSquare = ((v + h) % 2 == 0);  
            Color squareColor = isWhiteSquare ? new Color(null, 255, 255,255) : new Color(null, 0, 192, 0);  
  
            gc.setBackground(squareColor);  
            gc.fillRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);  
  
            gc.setForeground(new Color(null, 0, 0, 0));  
            gc.drawRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);  
        } // for  
    e.gc.drawRect(0, 0, clientArea.width - 1, clientArea.height - 1);  
} // paintControl
```

Шахматы. Создание поля (field) для необъявленных **nV** и **nH**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



```
16 }
17
18 @Override
19 public void paintControl(PaintEvent e) {
20     Rectangle clientArea = getClientArea();
21     GC gc = e.gc;
22     for (int v = 0; v < nV; v++)
23         for (int h = 0; h < nH; h++) {
24             int squareWidth
25             int squareHeight
26
27             boolean isWhiteSquare
28             Color squareColor
29
30             gc.setBackground
31             gc.fillRect
32
33             gc.setForeground
34             gc.drawRect
35         } // for
36     e.gc.drawRect(0, 0, clientArea.width - 1, clientArea.height - 1);
```

nH cannot be resolved to a variable

6 quick fixes available:

- Create local variable 'nH'
- Create field 'nH'
- Change to 'h'
- Create parameter 'nH'
- Create constant 'nH'
- Change to 'v'

Шахматы. Отрисовка клеток доски

МГУ им. М.В.Ломоносова. Факультет ВМК.

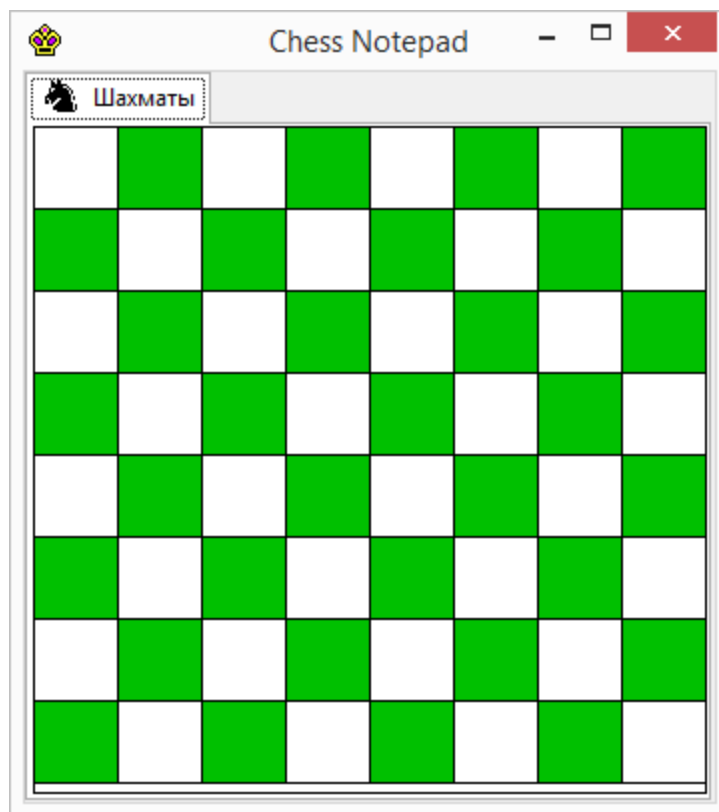
Романов Владимир Юрьевич ©2025

```
Chess.java ChessBoard.java ✕
18 private int nV = 8;
19 private int nH = 8;
20
21 @Override
22 public void paintControl(PaintEvent e) {
23     Rectangle clientArea = getClientArea();
24     GC gc = e.gc;
25     for (int v = 0; v < nV; v++)
26         for (int h = 0; h < nH; h++) {
27             int squareWidth = getClientArea().width / nV;
28             int squareHeight = getClientArea().height / nH;
29
30             boolean isWhiteSquare = ((v + h) % 2 == 0);
31             Color squareColor = isWhiteSquare ? new Color(null, 255, 255, 255) : new Color(null, 0, 192, 0);
32
33             gc.setBackground(squareColor);
34             gc.fillRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);
35
36             gc.setForeground(new Color(null, 0, 0, 0));
37             gc.drawRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);
38         } // for
39     e.gc.drawRect(0, 0, clientArea.width - 1, clientArea.height - 1);
40 }
41 }
```

Шахматы. Клетки шахматной доски

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Рефакторинг программы. Разновидности досок для игр

Рефакторинг. Вынос универсальных полей и методов в базовый класс

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

❖ Предпосылки

- Размерность доски - поля nH и nV общие для всех клеточных игр
- Алгоритм рисования в цикле всех клеток доски универсален

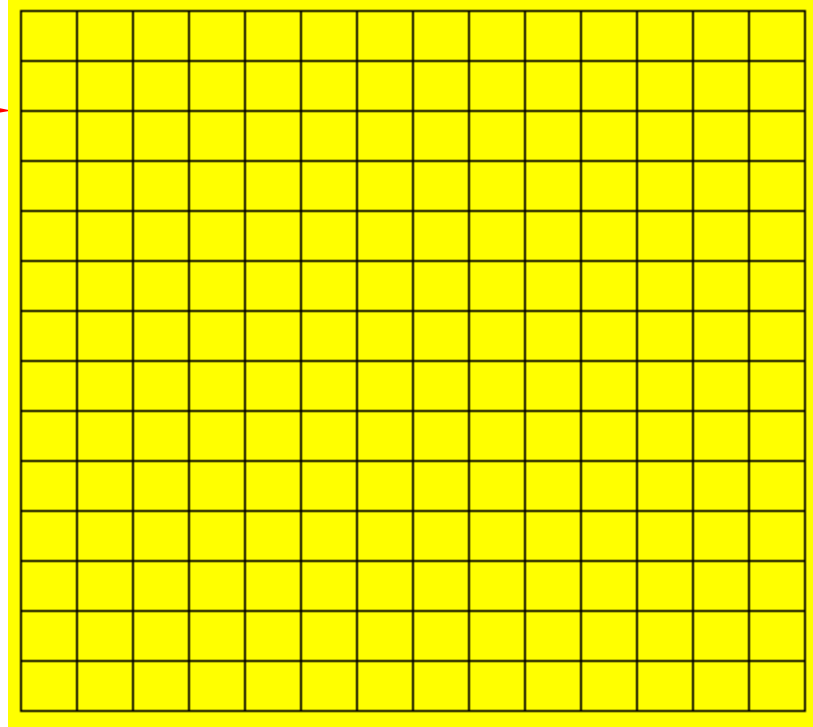
Доска для азиатских игр.

Клетки доски для игр го и рендзю

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Фигуры
располагаются на
пересечении линий



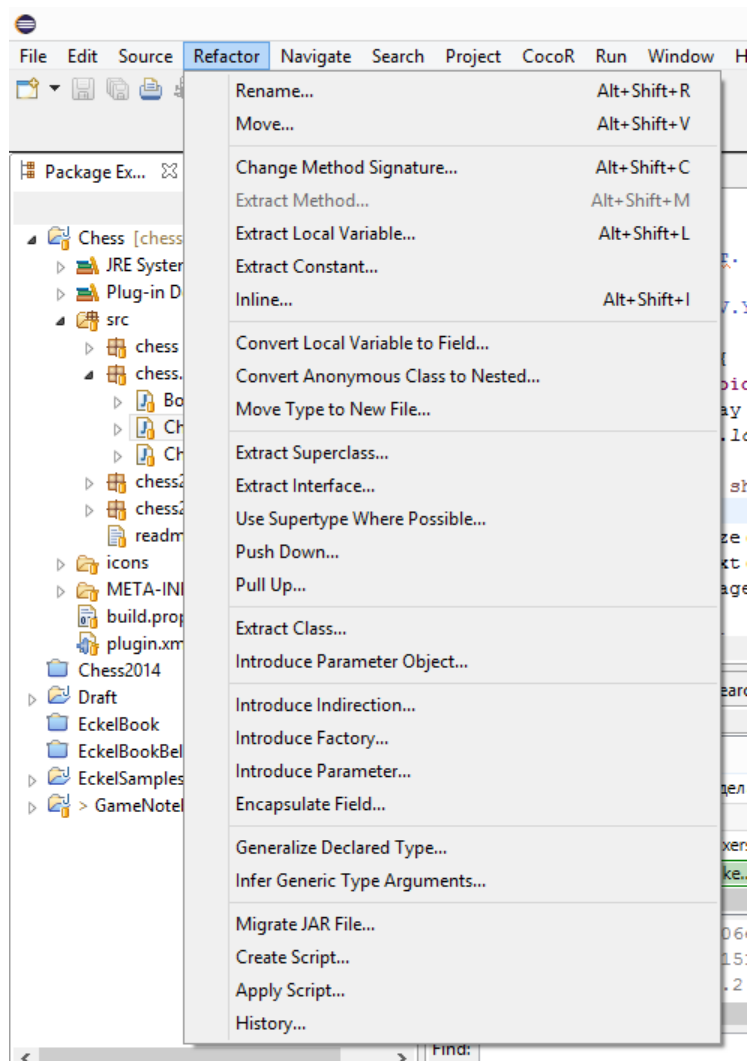
- Цель – преобразовать программу для рисования досок различных настольных игр

Рефакторинг.

Инструменты рефакторинга

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



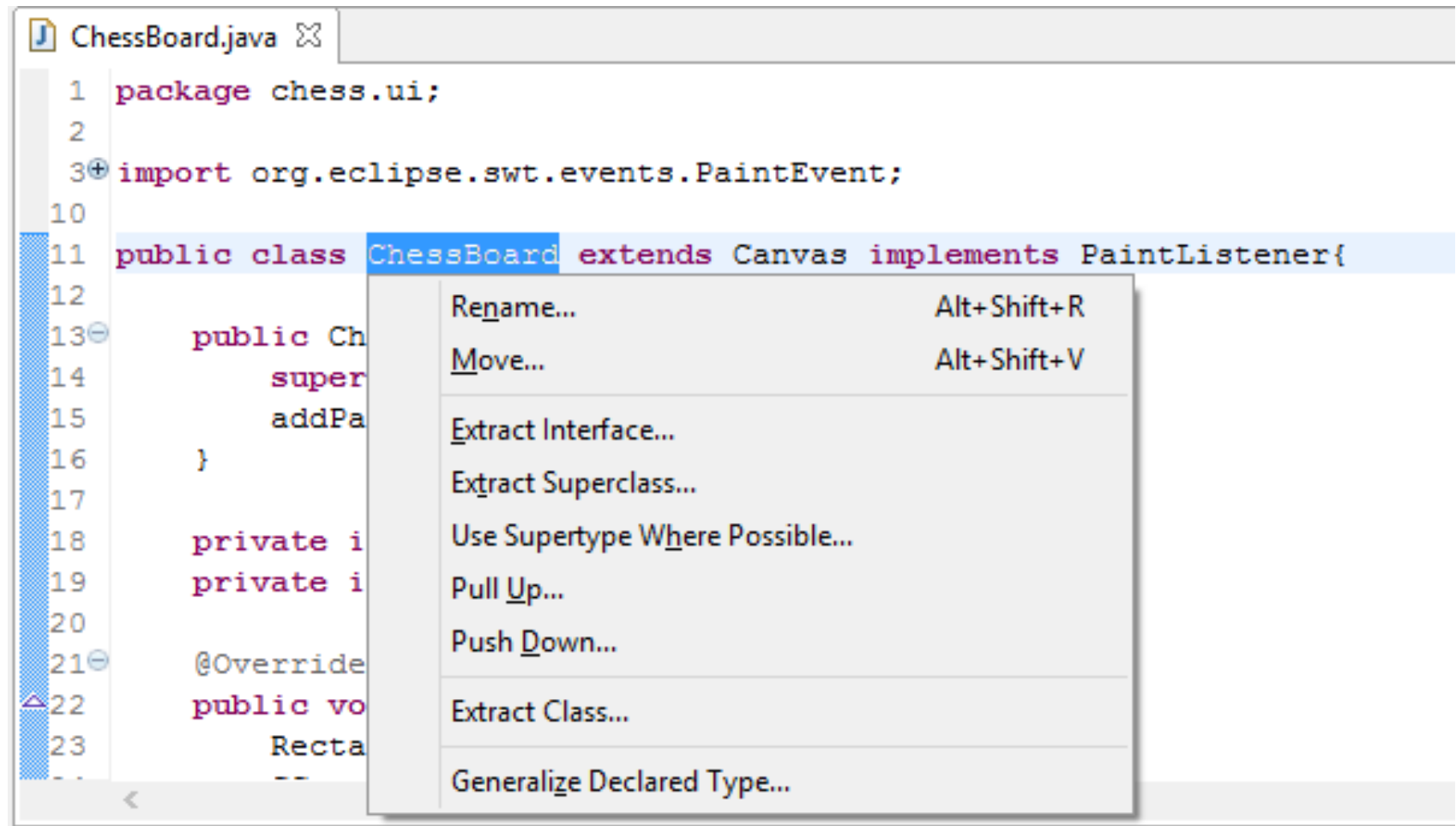
Контексто-зависимый рефакторинг.

Клавиши **Alt + Shift + T** (Eclipse)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Преобразования применимые к классу **ChessBoard**



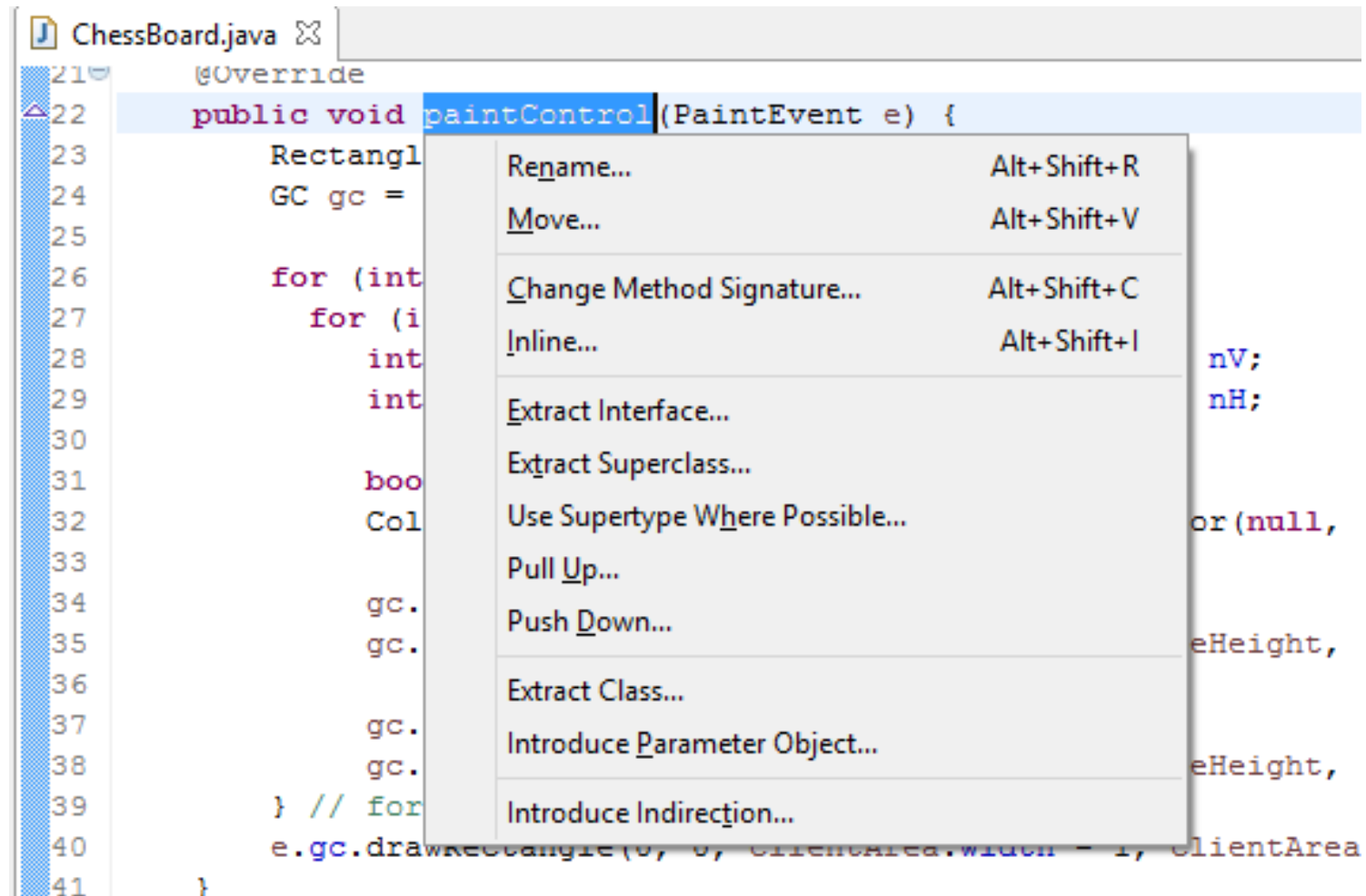
Контексто-зависимый рефакторинг.

Клавиши **Alt + Shift + T**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Преобразования применимые к методу **paintControl**



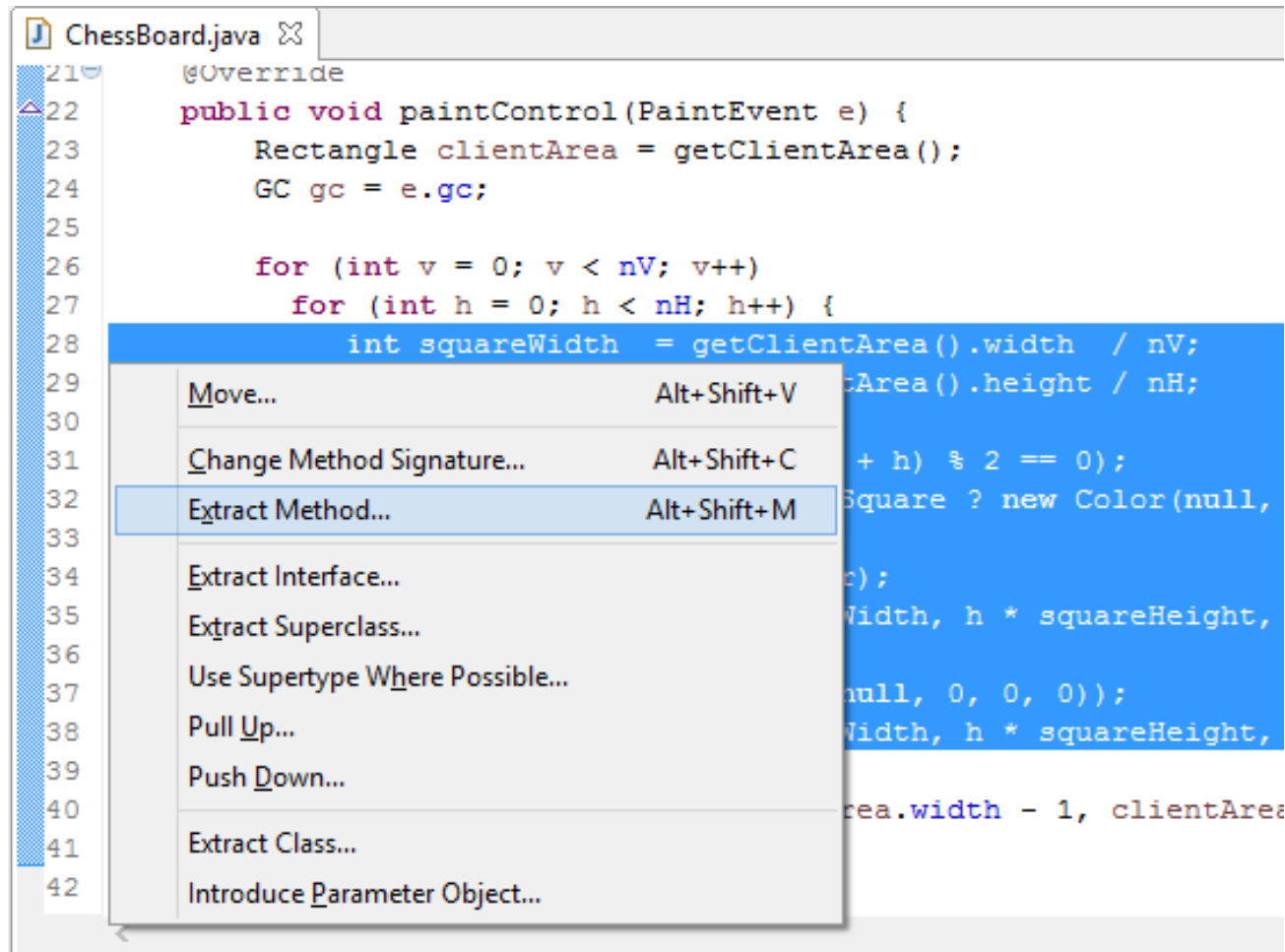
Контексто-зависимый рефакторинг.

Клавиши **Alt + Shift + T**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Преобразования применимые к выделенному тексту



Шахматы. Отрисовка клеток доски

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public void paintControl(PaintEvent e) {  
    Rectangle clientArea = getClientArea();  
    GC gc = e.gc;  
    for (int v = 0; v < nV; v++)  
        for (int h = 0; h < nH; h++) {
```

```
        int squareWidth = getClientArea().width / nV;  
        int squareHeight = getClientArea().height / nH;
```

```
        boolean isWhiteSquare = ((v + h) % 2 == 0);
```

```
        Color squareColor = isWhiteSquare ? new Color(null, 255, 255,255) : new Color(null, 0, 192, 0);
```

```
        gc.setBackground(squareColor);
```

```
        gc.fillRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);
```

```
        gc.setForeground(new Color(null, 0, 0, 0));
```

```
        gc.drawRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);
```

```
    } // for
```

```
    e.gc.drawRect(0, 0, clientArea.width - 1, clientArea.height - 1);
```

```
    } // paintControl
```

Алгоритм рисования клетки
шахматной доски вынесем в
отдельный метод

Рефакторинг. Выделение метода.

Клавиши **ALT + Shift + M**

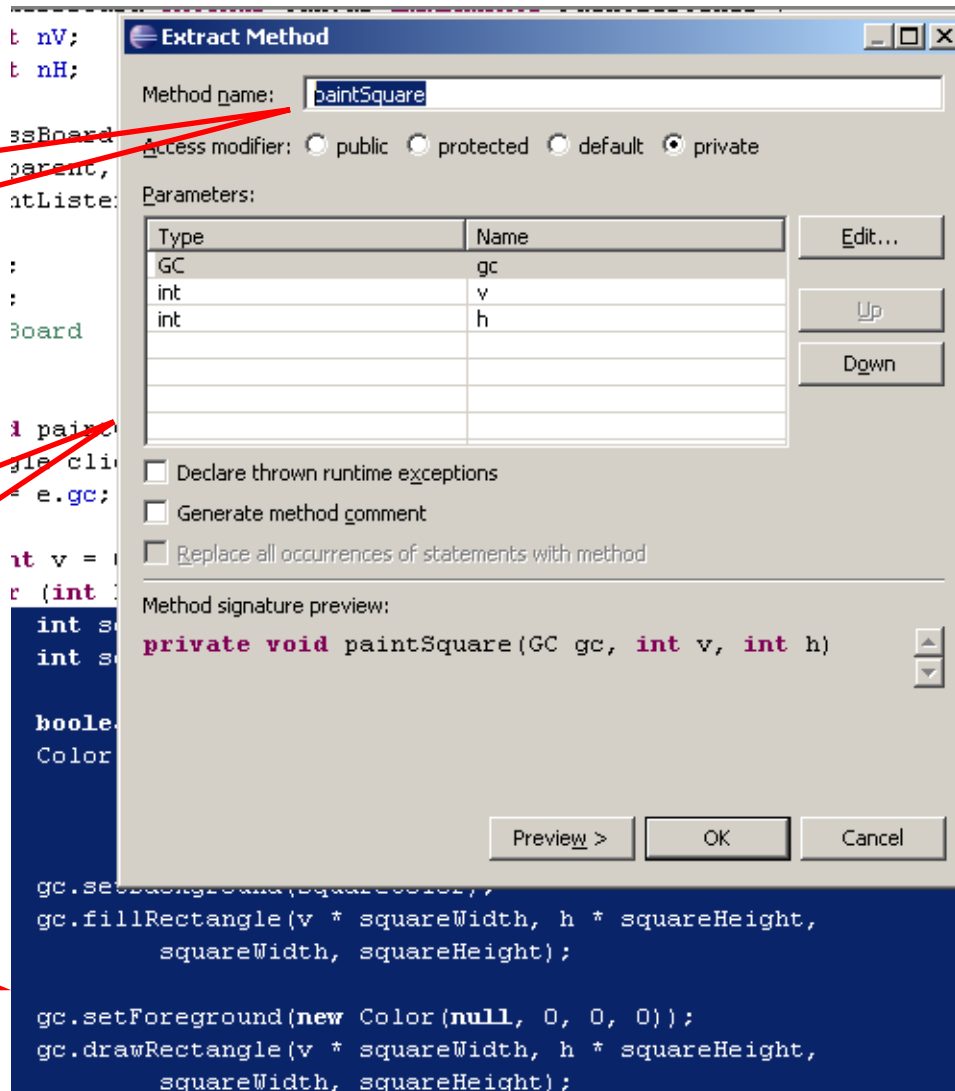
МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

3. напечатать
имя метода

2. Потом открыть
диалоговое окно
ALT + Shift + M

1. Сначала
выделение
текста



Шахматы. Выделенный метод рисования клетки шахмат (двухцветная доска)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
private void drawSquare(GC gc, int v, int h) {  
    int squareWidth = getClientArea().width / nV;  
    int squareHeight = getClientArea().height / nH;  
  
    boolean isWhiteSquare = ((v + h) % 2 == 0);  
    Color squareColor = isWhiteSquare  
        ? new Color(null, 255, 255,255) : new Color(null, 0, 192, 0);  
  
    gc.setBackground(squareColor);  
    gc.fillRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);  
  
    gc.setForeground(new Color(null, 0, 0, 0));  
    gc.drawRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);  
}
```

Шахматы. Вызов выделенного метода в универсальном методе рисования доски

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public void paintControl(PaintEvent e) {  
    Rectangle clientArea = getClientArea();  
  
    GC gc = e.gc;  
  
    for (int v = 0; v < nV; v++)  
        for (int h = 0; h < nH; h++) {  
            drawSquare(gc, v, h);  
        }  
  
    gc.drawRectangle(0, 0, clientArea.width - 1, clientArea.height - 1);  
}
```

Рефакторинг. Вынос универсальных полей и методов в базовый класс

МГУ им. М.В.Ломоносова. Факультет ВМК.

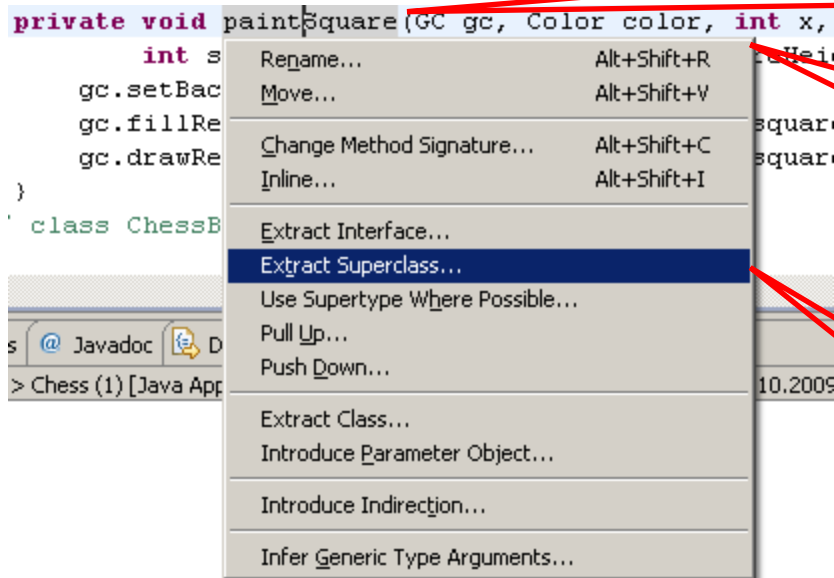
Романов Владимир Юрьевич ©2025

- Предпосылки
 - Размерность доски - поля nH и nV общие для всех клеточных игр
 - Алгоритм рисования в цикле всех клеток доски универсален

Рефакторинг. Создание абстрактного класса *GameBoard*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



1. Сначала
выделение имени

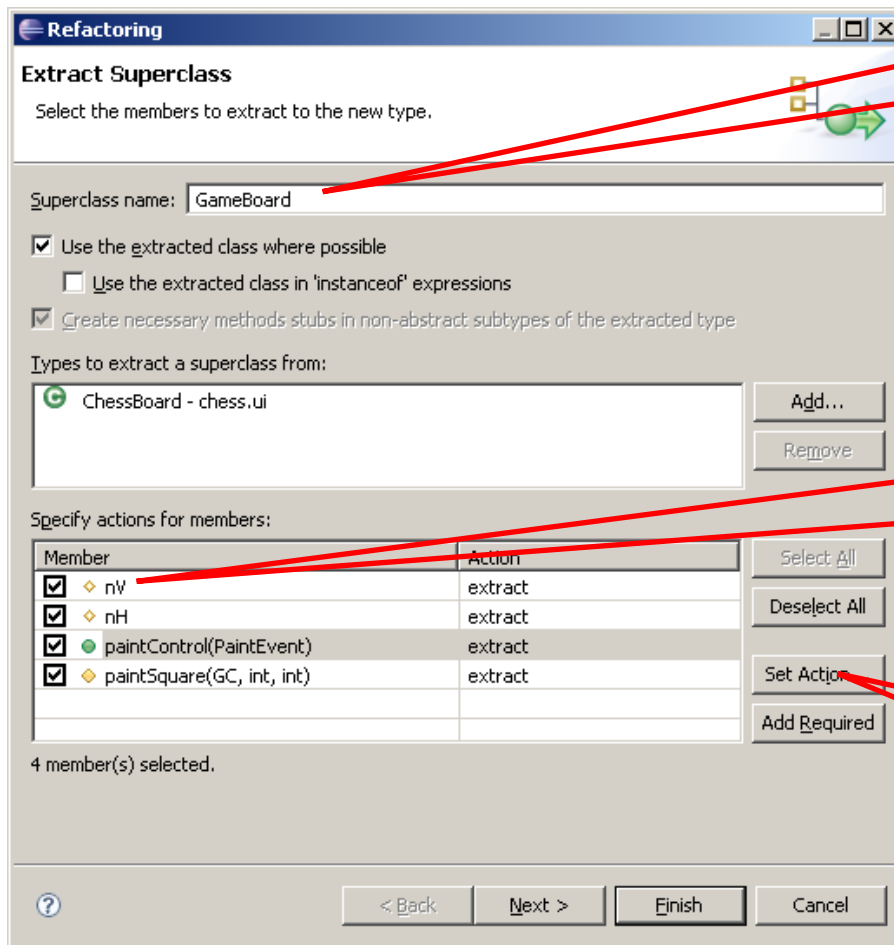
2. Переход к меню
допустимых рефакторингов
ALT + Shift + T

3. Выбор
рефакторинга
Extract Superclass

Рефакторинг. Создание абстрактного класса *GameBoard*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



1. Ввод имени суперкласса

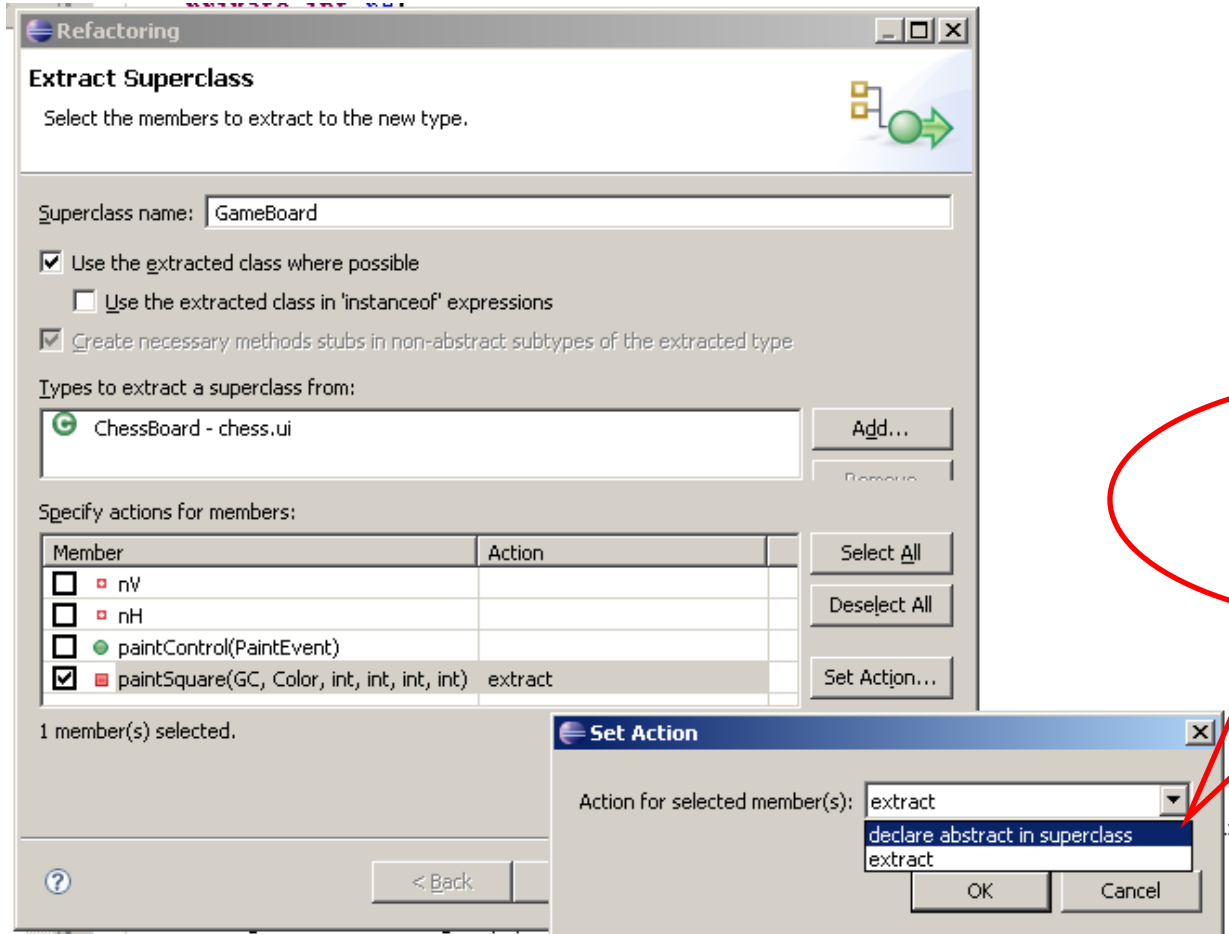
2. Выбор переносимых в суперкласс полей и методов

3. Выбор действия умалчиваемое действие **extract**

Рефакторинг. Создание абстрактного метода в классе *GameBoard*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Шахматы. Выделенный суперкласс

GameBoard

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class GameBoard extends Canvas {  
    protected int nV;  
    protected int nH;  
    protected abstract void paintSquare(GC gc, int v, int h);  
  
    public GameBoard(Composite parent, int style)  
        { super(parent, style); }  
  
    public void paintControl(PaintEvent e) {  
        Rectangle clientArea = getClientArea();  
        GC gc = e.gc;  
        for (int v = 0; v < nV; v++)  
            for (int h = 0; h < nH; h++)  
                paintSquare(gc, v, h);  
        e.gc.drawRectangle(0, 0, clientArea.width - 1, clientArea.height - 1);  
    } // paintControl  
}
```

Предок
выделенного
суперкласса

Переопределяемый
метод *paintSquare*

Шахматы. *ChessBoard* – потомок выделенного суперкласса *GameBoard*

МГУ им. М.В.Ломоносова. Факультет ВМК.

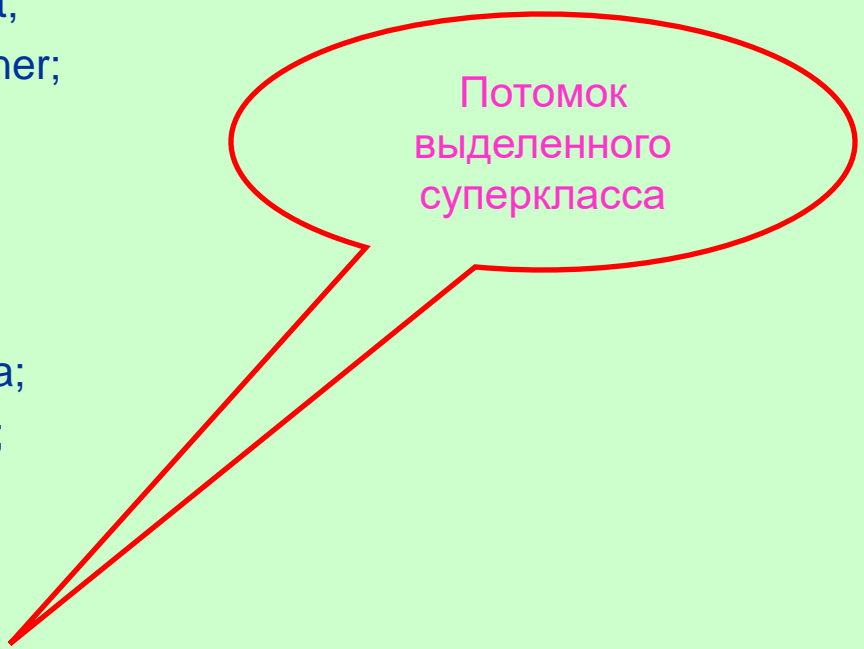
Романов Владимир Юрьевич ©2025

```
import org.eclipse.swt.events.MouseEvent;
import org.eclipse.swt.events.MouseListener;
import org.eclipse.swt.graphics.Color;
import org.eclipse.swt.graphics.Cursor;
import org.eclipse.swt.graphics.GC;
import org.eclipse.swt.graphics.Image;
import org.eclipse.swt.graphics.ImageData;
import org.eclipse.swt.widgets.Composite;

import chess.ChessImages;

public class ChessBoard extends GameBoard {

    // ...
}
```



Потомок
выделенного
суперкласса

Перенос реализации интерфейса MouseListener в базовый класс

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
ChessBoard.java GameBoard.java X
1 package chess.ui;
2
3+ import org.eclipse.swt.events.PaintEvent;
9
10 public abstract class GameBoard extends Canvas implements PaintListener {
11-     public GameBoard(Composite parent, int style) {
12         super(parent, style);
13         addPaintListener(this);
14     }
--
```

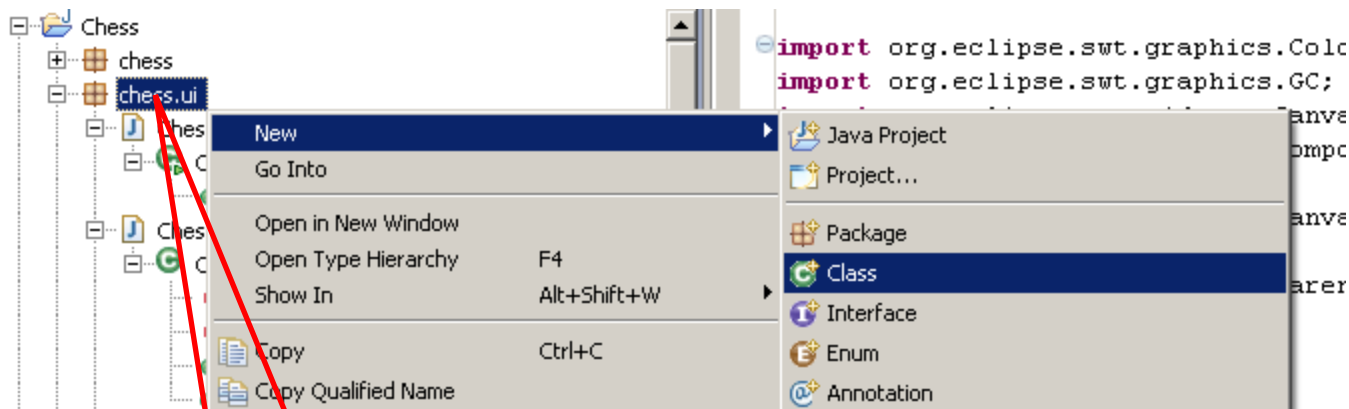


```
ChessBoard.java X GameBoard.java
4 import org.eclipse.swt.graphics.Color;
5 import org.eclipse.swt.graphics.GC;
6 import org.eclipse.swt.widgets.Composite;
7
8 public class ChessBoard extends GameBoard implements PaintListener {
9
10-     public ChessBoard(Composite parent, int style) {
11         super(parent, style);
12         addPaintListener(this);
13     }
14
```

Создание класса AsiaBoard - потомка класса *GameBoard* (wizard)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Создание класса в
пакете chess.ui

Создание класса *AsiaBoard* - потомка класса *GameBoard* (wizard)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Задание имени
класса

Задание имени
базового класса

New Java Class

Create a new Java class.

Source folder: Chess/src Browse...

Package: chess.ui Browse...

☐ Enclosing type: Browse...

Name: AsiaBoard

Modifiers: ☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static

Superclass: GameBoard Browse...

Interfaces: Add...

Which method stubs would you like to create?

☐ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

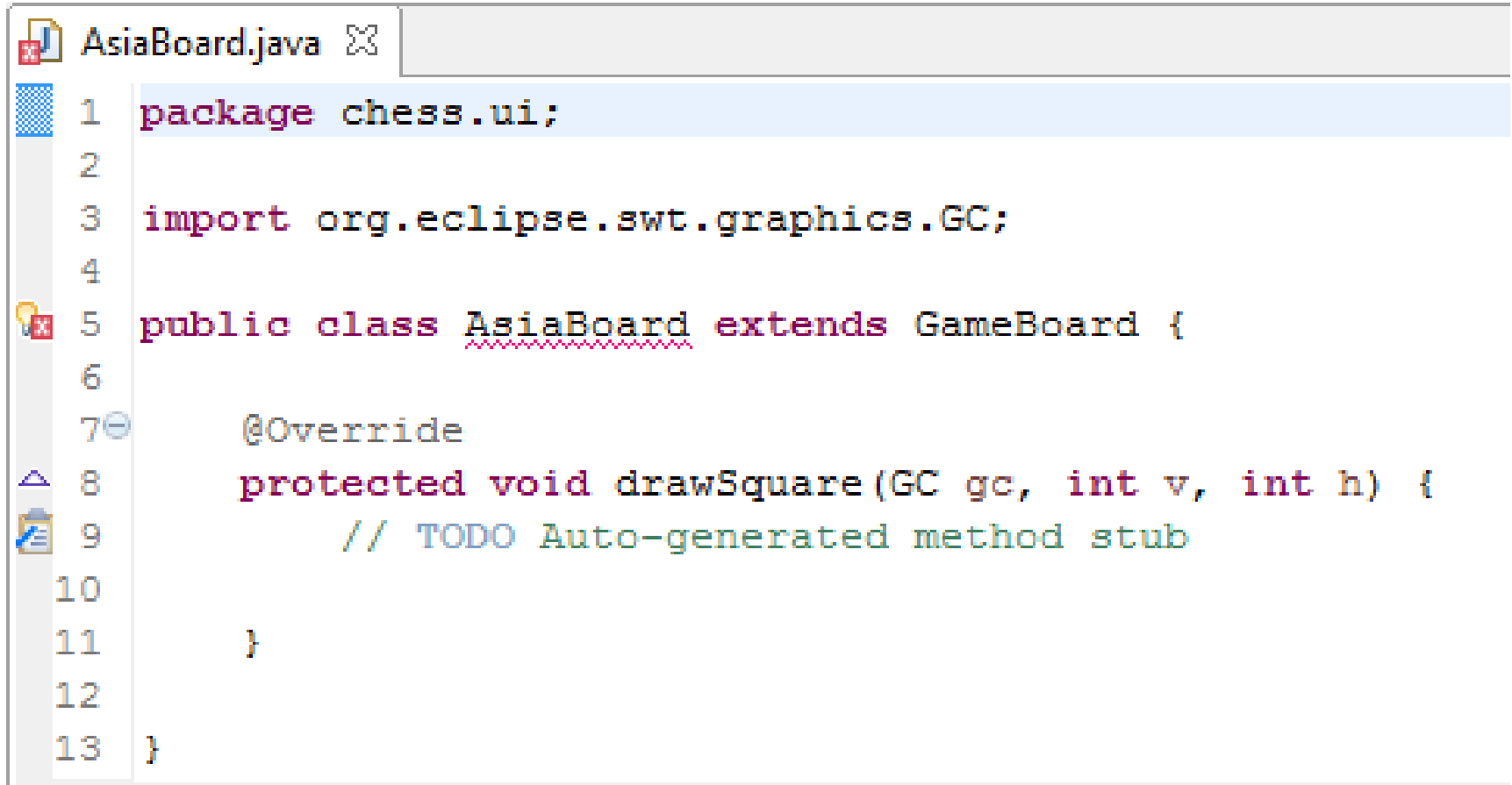
Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

Finish Cancel

Создан класс *AsiaBoard*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

A screenshot of a Java IDE window titled 'AsiaBoard.java'. The code is as follows:

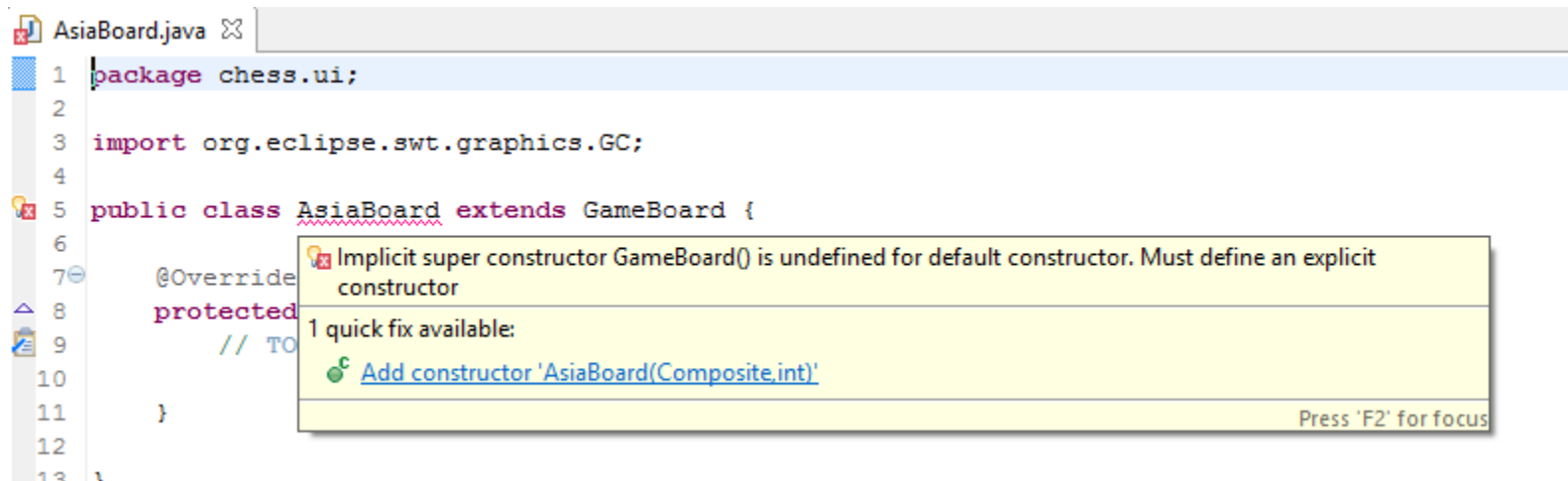
```
1 package chess.ui;  
2  
3 import org.eclipse.swt.graphics.GC;  
4  
5 public class AsiaBoard extends GameBoard {  
6  
7     @Override  
8     protected void drawSquare(GC gc, int v, int h) {  
9         // TODO Auto-generated method stub  
10  
11     }  
12  
13 }
```

The code is displayed with standard Java syntax highlighting: keywords in purple, package names in blue, and comments in green. The class name 'AsiaBoard' is underlined with a red squiggly line, indicating it is not yet defined. The IDE interface includes a left sidebar with icons for Explorer, Search, and Run and Debug, and a top toolbar with icons for file operations and running the program.

Создание конструктора класса *AsiaBoard*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



```
AsiaBoard.java
1 package chess.ui;
2
3 import org.eclipse.swt.graphics.GC;
4
5 public class AsiaBoard extends GameBoard {
6
7     @Override
8     protected
9         // TO
10
11 }
12
13
```

Implicit super constructor GameBoard() is undefined for default constructor. Must define an explicit constructor

1 quick fix available:

- Add constructor 'AsiaBoard(Composite,int)'

Press 'F2' for focus

Конструктор класса *AsiaBoard* создан

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
AsiaBoard.java ✕  
1 package chess.ui;  
2  
3 import org.eclipse.swt.graphics.GC;  
4 import org.eclipse.swt.widgets.Composite;  
5  
6 public class AsiaBoard extends GameBoard {  
7  
8     public AsiaBoard(Composite parent, int style) {  
9         super(parent, style);  
10    }  
11  
12    @Override  
13    protected void drawSquare(GC gc, int v, int h) {  
14        // TODO Auto-generated method stub  
15    }  
16 }
```

Шахматы. *AsiaBoard* – потомок суперкласса *GameBoard* (1)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

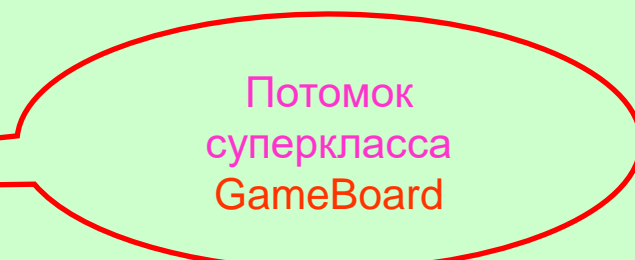
```
package chess.ui;

import org.eclipse.swt.graphics.GC;
import org.eclipse.swt.widgets.Composite;

public class AsiaBoard extends GameBoard {

    public AsiaBoard(Composite parent, int style) {
        super(parent, style);
    }

    @Override
    protected void drawSquare(GC gc, int v, int h) {
        // TODO Auto-generated method stub
    }
}
```



Потомок
суперкласса
GameBoard

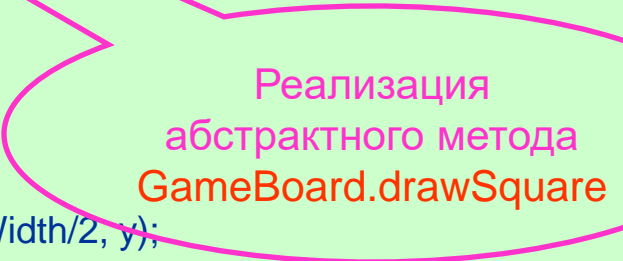
Шахматы. *AsiaBoard* – потомок суперкласса *GameBoard* (2)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

@Override

```
protected void drawSquare(GC gc, int v, int h) {  
    int squareWidth = getClientArea().width / nV;  
    int squareHeight = getClientArea().height / nH;  
  
    gc.setBackground(new Color(null, 255, 255, 0));  
    gc.fillRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);  
  
    int x = v * squareWidth + squareWidth/2;  
    int y = h * squareHeight + squareHeight/2;  
  
    if (v != 0) gc.drawLine(x, y, x - squareWidth/2, y);  
    if (v != nV-1) gc.drawLine(x, y, x + squareWidth/2, y);  
  
    if (h != 0) gc.drawLine(x, y, x, y - squareHeight/2);  
    if (h != nH-1) gc.drawLine(x, y, x, y + squareHeight/2);  
}  
}
```



Реализация
абстрактного метода
GameBoard.drawSquare

Блокнот игр. Доска для игр:

Шахматы, Рендзю

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

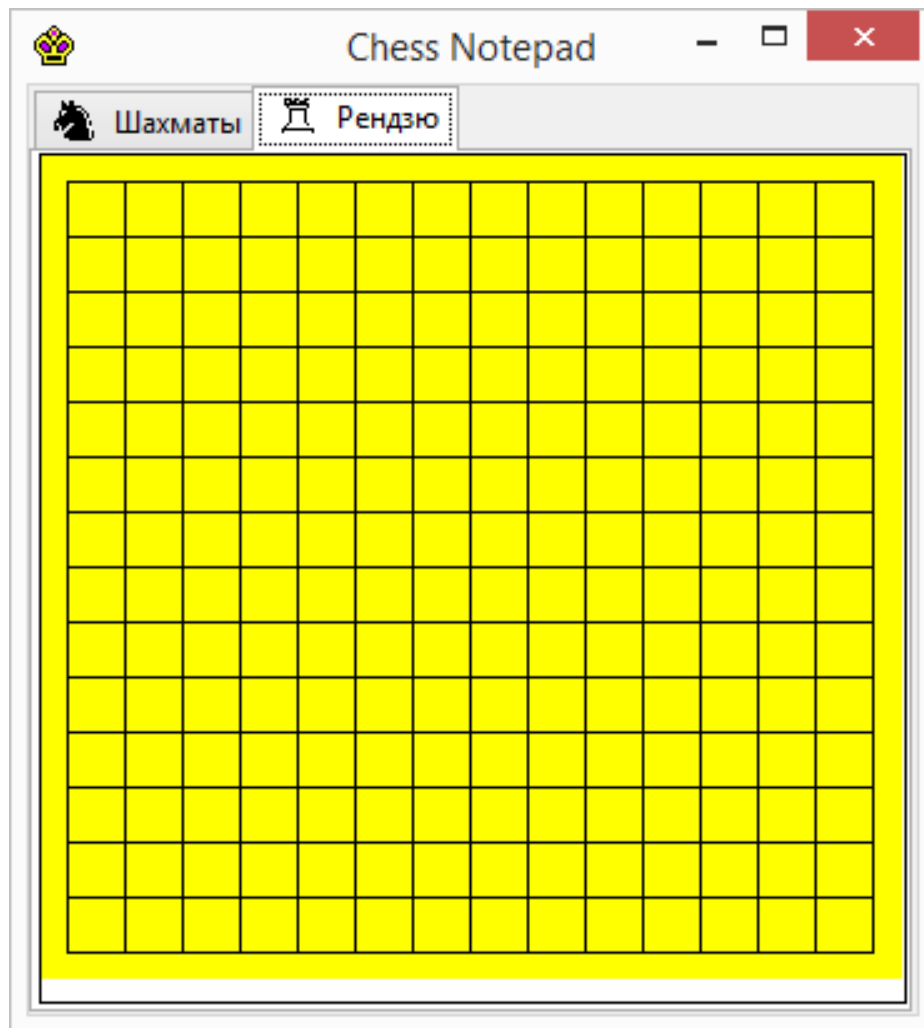
```
AsiaBoard.java  Chess.java ✕
29      FillLayout layout = new FillLayout();
30      shell.setLayout(layout);
31
32      TabFolder gamesFolder = new TabFolder(shell, SWT.TOP);
33
34      Image chessImage = new Image(display, ChessImages.imageKnightBlack
35          .getImageData().scaledTo(20, 20));
36
37      TabItem chessItem = new TabItem(gamesFolder, SWT.NONE);
38      chessItem.setText("Шахматы");
39      chessItem.setControl(new ChessBoard(gamesFolder, SWT.NONE));
40      chessItem.setImage(chessImage);
41
42      final Image renjuTabImage = new Image(display,
43          ChessImages.imageRookWhite.getImageData().scaledTo(20, 20));
44
45      TabItem renjuItem = new TabItem(gamesFolder, SWT.NULL);
46      renjuItem.setText("Рендзю");
47      renjuItem.setImage(renjuTabImage);
48      renjuItem.setControl(new AsiaBoard(gamesFolder, SWT.NONE));
49
```

Блокнот игр. Доска для игр:

Шахматы, Рендзю

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Блокнот игр. Игры на шахматной доске

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
final TabFolder gamesFolder = new TabFolder(shell, SWT.TOP);
```

```
final Image chessTabImage = new Image(display,  
    ChessImages.imageKnightBlack.getImageData().scaledTo(20, 20));
```

Блокнот

```
TabItem chessItem = new TabItem(gamesFolder, SWT.NULL);  
chessItem.setText("Шахматы");  
chessItem.setImage(chessTabImage);  
chessItem.setControl(new ChessBoard(gamesFolder, SWT.NONE));
```

```
final Image checkersTabImage = new Image(display,  
    ChessImages.imageCheckersNotebook.getImageData().scaledTo(20, 20));
```

```
TabItem checkersItem = new TabItem(gamesFolder, SWT.NULL);  
checkersItem.setText("Шашки");  
checkersItem.setImage(checkersTabImage);  
checkersItem.setControl(new ChessBoard(gamesFolder, SWT.NONE));
```


Блокнот игр. Игры на «японской» доске

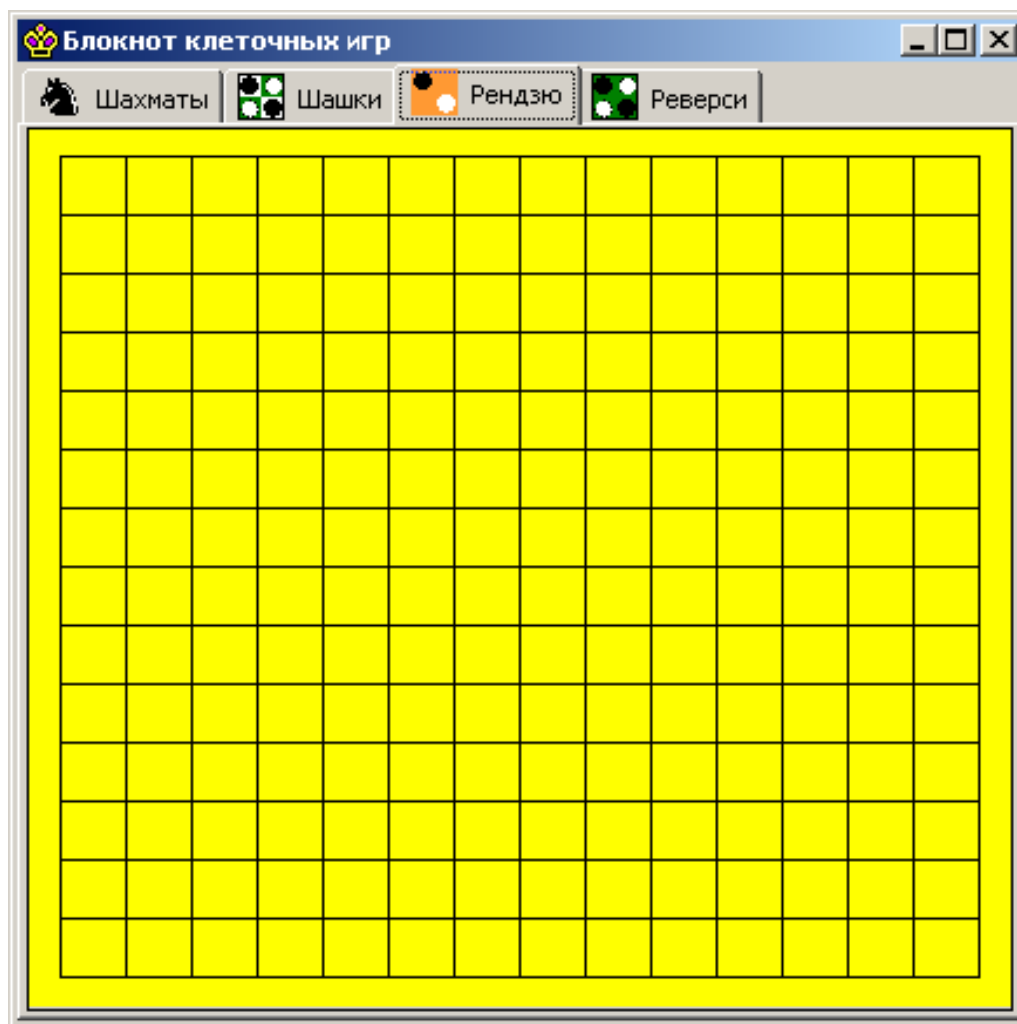
```
final Image renjuTabImage = new Image(display,  
    ChessImages.imageRenjuNotebook.getImageData().scaledTo(20, 20));  
  
TabItem renjuItem = new TabItem(gamesFolder, SWT.NULL);  
renjuItem.setText("Рендзю");  
renjuItem.setImage(renjuTabImage);  
renjuItem.setControl(new AsiaBoard(gamesFolder, SWT.NONE));  
  
final Image reversiTabImage = new Image(display,  
    ChessImages.imageReversiNotebook.getImageData().scaledTo(20, 20));  
  
TabItem reversiItem = new TabItem(gamesFolder, SWT.NULL);  
reversiItem.setText("Реверси");  
reversiItem.setImage(reversiTabImage);  
reversiItem.setControl(new AsiaBoard(gamesFolder, SWT.NONE));
```

Блокнот игр. Доска для игры

Рендзю, Го, Сянци (китайские шахматы)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



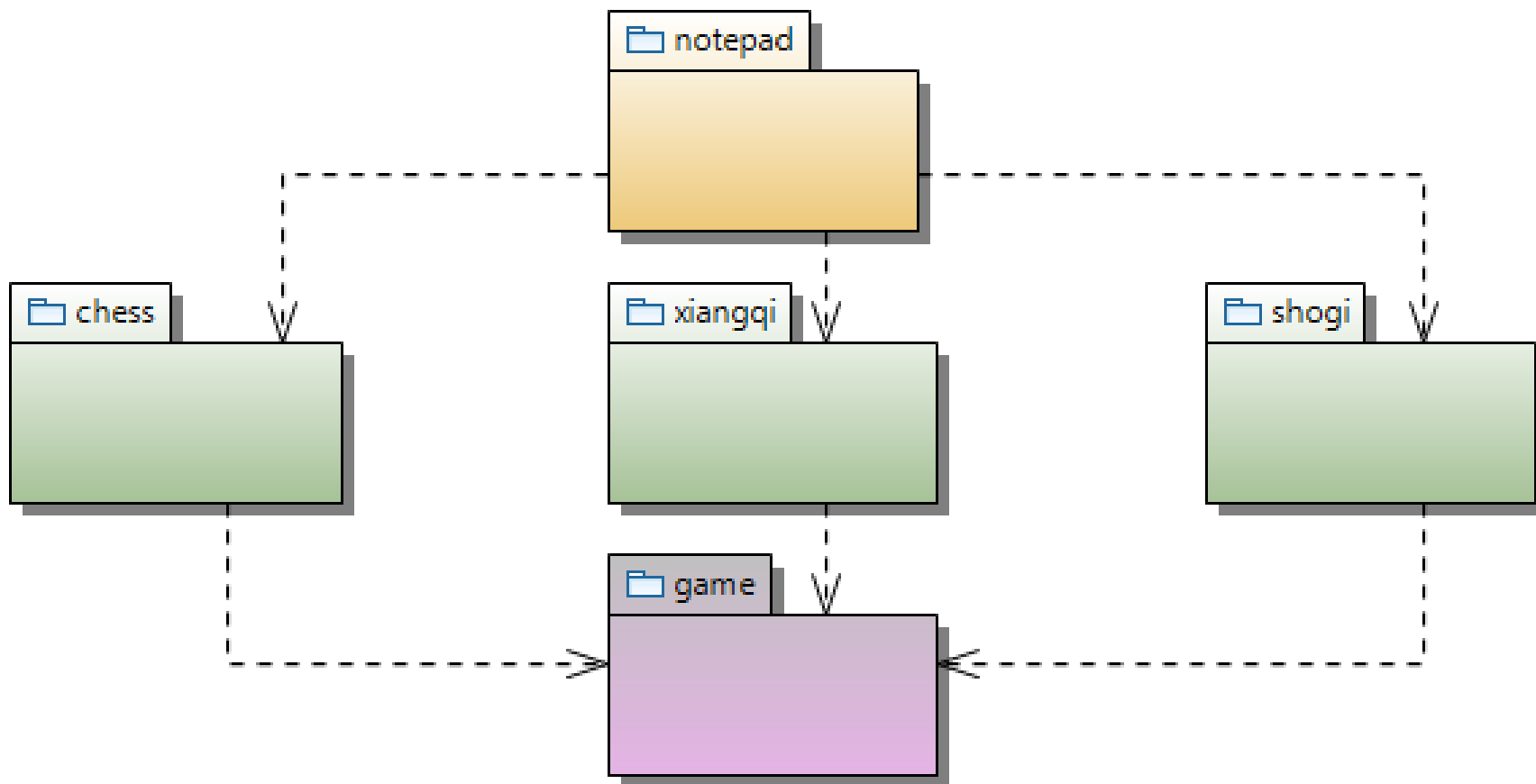
Проектирование архитектуры.

Выделение уровней в настольных играх

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

Идентификация *пакетов верхнего уровня*

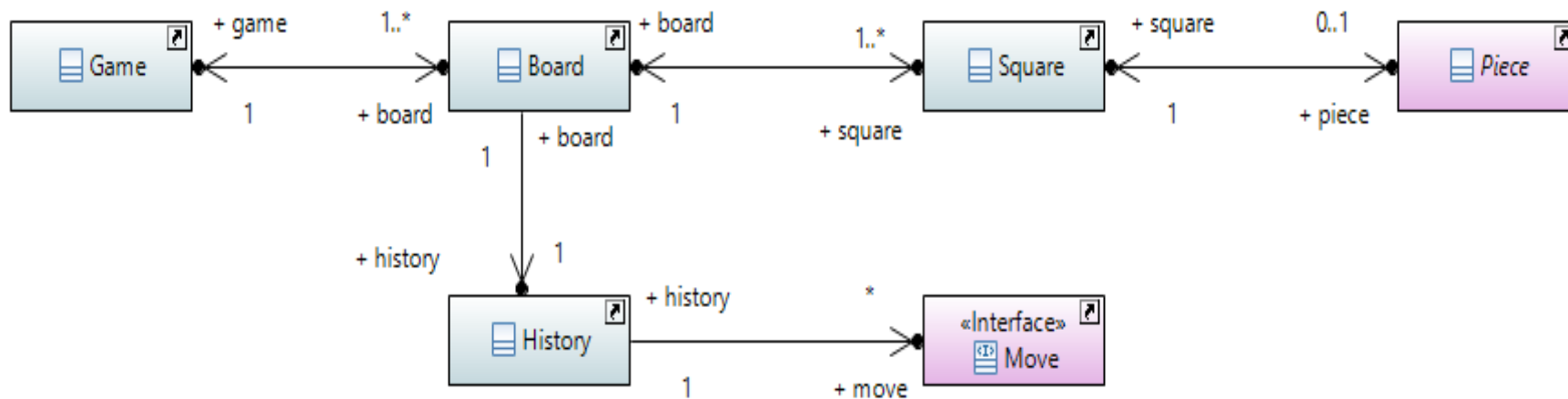


Проектирование архитектуры.

Классы пакета **game.core**

МГУ им. М.В.Ломоносова. Факультет ВМК.

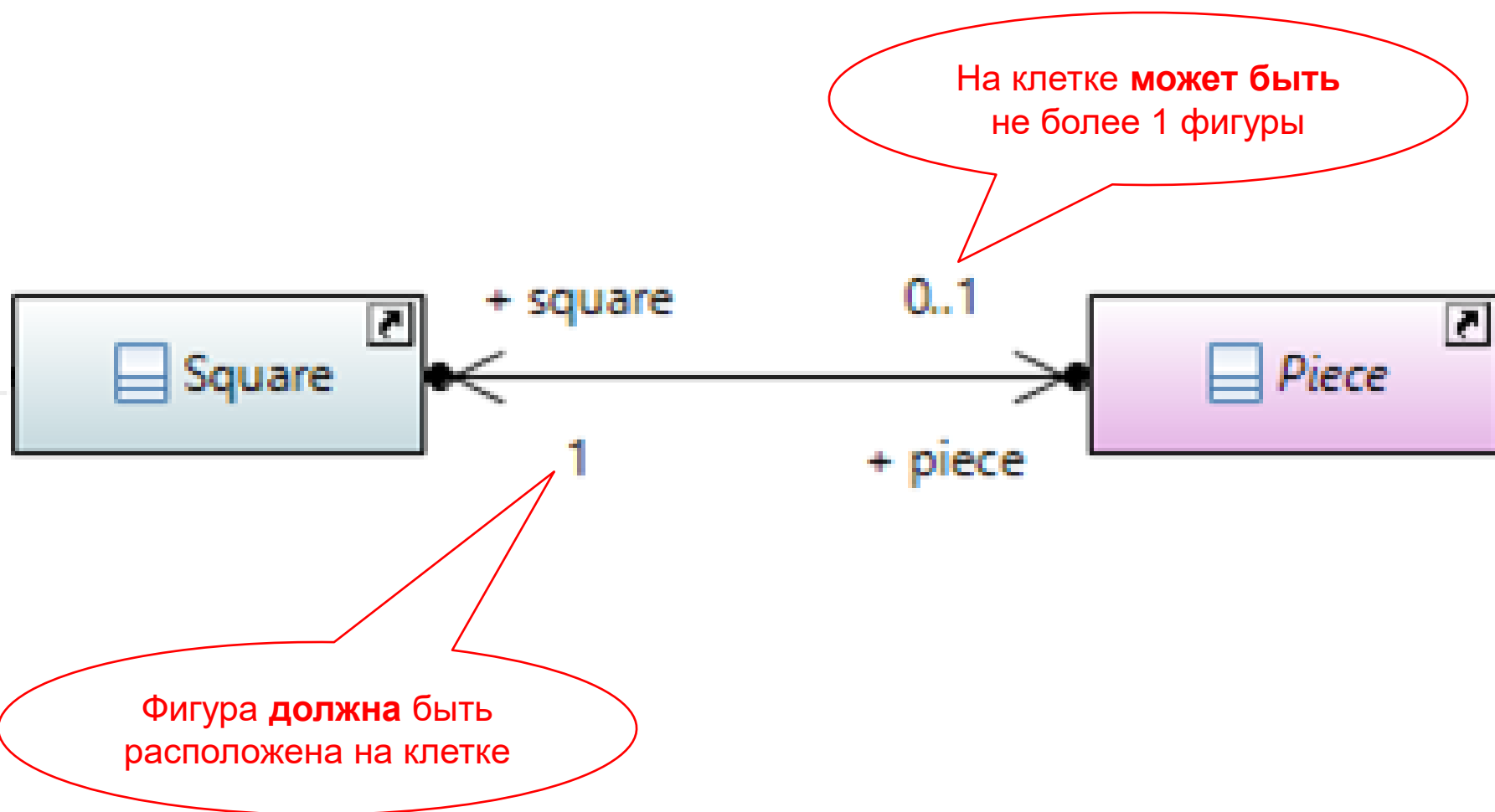
Романов Владимир Юрьевич ©2025



Отношение ассоциации между классам *Piece* и *Square*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Перечисление *PieceColor* - цвет фигуры

```
package game.core;  
  
public enum PieceColor {  
    WHITE,  
    BLACK,  
    GREEN,  
    BLUE  
}
```

Класс Piece (фигура)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
abstract public class Piece {  
    private Square square;  
    PieceColor color;  
  
    public Piece(Square square, PieceColor color) {  
        setSquare(square);  
        this.color = color;  
    }  
  
    public Square getSquare() {  
        return square;  
    }  
  
    public void setSquare(Square square) {  
        this.square = square;  
        square.piece = this;  
    }  
    // ...  
}
```

Фигура **должна** быть
расположена на клетке

Класс Square (клетка)

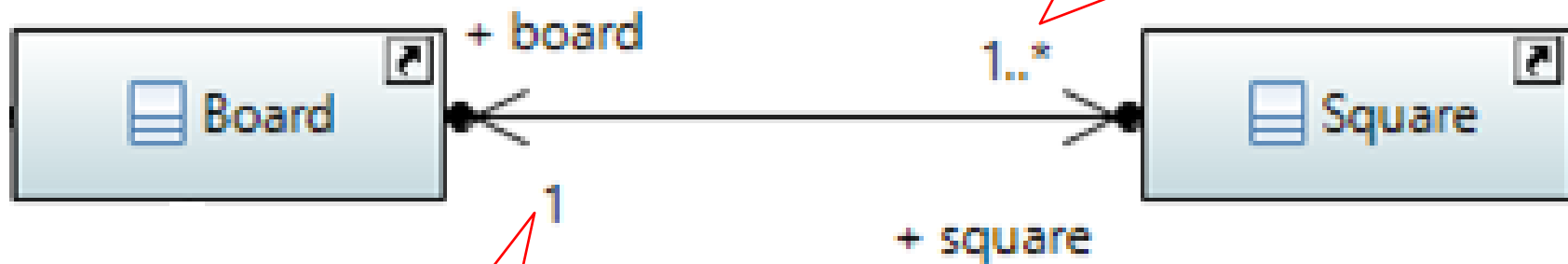
```
public class Square {  
    public Piece piece;  
  
    public Board board;  
    public int v;  
    public int h;  
  
    public Square(Board board, int v, int h) {  
        this.board = board;  
        this.v = v;  
        this.h = h;  
    }  
  
    public Piece getPiece() {  
        return piece;  
    }  
  
    public void putPiece(Piece piece) {  
        this.piece = piece;  
        piece.setSquare(this);  
    }  
}
```

На клетке **может быть**
не более 1 фигуры

Отношение ассоциации между классам *Board* и *Square*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

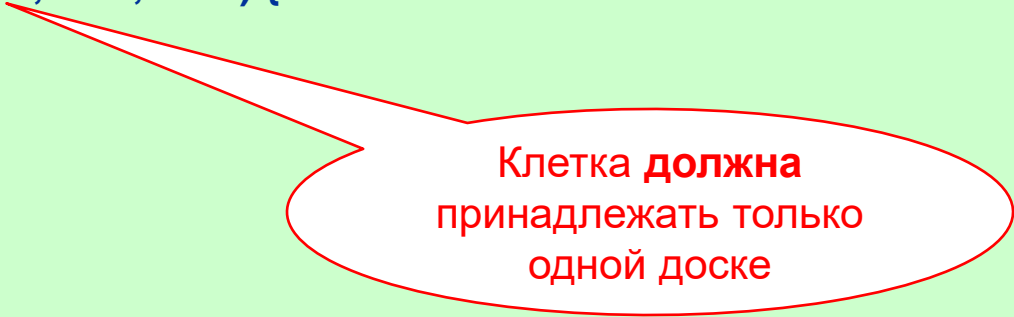


На доске может быть
1 и более клеток

Клетка должна
принадлежать только
одной доске

Класс Square (клетка)

```
public class Square {  
    public Piece piece;  
  
    public Board board;  
    public int v;  
    public int h;  
  
    public Square (Board board, int v, int h) {  
        this.board = board;  
        this.v = v;  
        this.h = h;  
    }  
  
    public Piece getPiece() {  
        return piece;  
    }  
  
    public Board getBoard() {  
        return board;  
    }  
}
```



Клетка **должна**
принадлежать только
одной доске

Класс Board (доска)

```
public class Board {
    private Square [ ][ ] squares;

    public Board(int nV, int nH) {
        squares = new Square[nV][nH];

        for (int v = 0; v < nV; v++)
            for (int h = 0; h < nH; h++)
                squares[v][h] = new Square(this, v, h);
    }

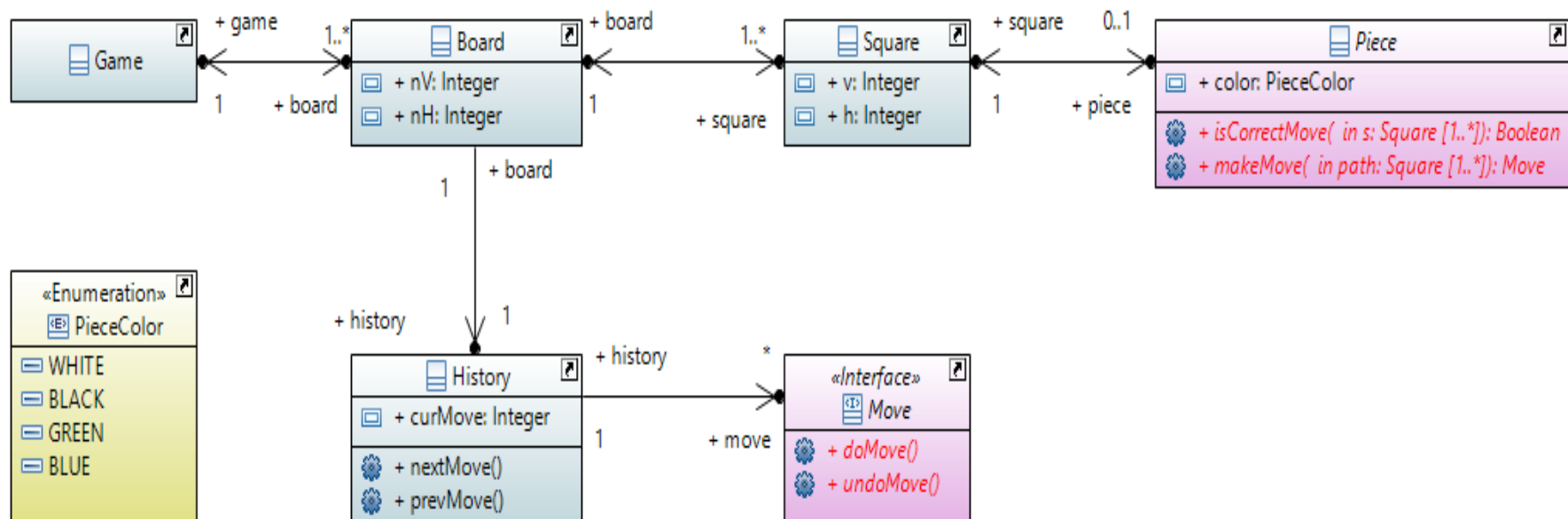
    public boolean isEmpty(int v, int h) {
        return getSquare(v, h).piece == null;
    }

    public Square getSquare(int v, int h) {
        return squares[v][h];
    }
}
```

Атрибуты и методы классов ядра настольных игр

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Блокнот игр. Шахматные фигуры

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

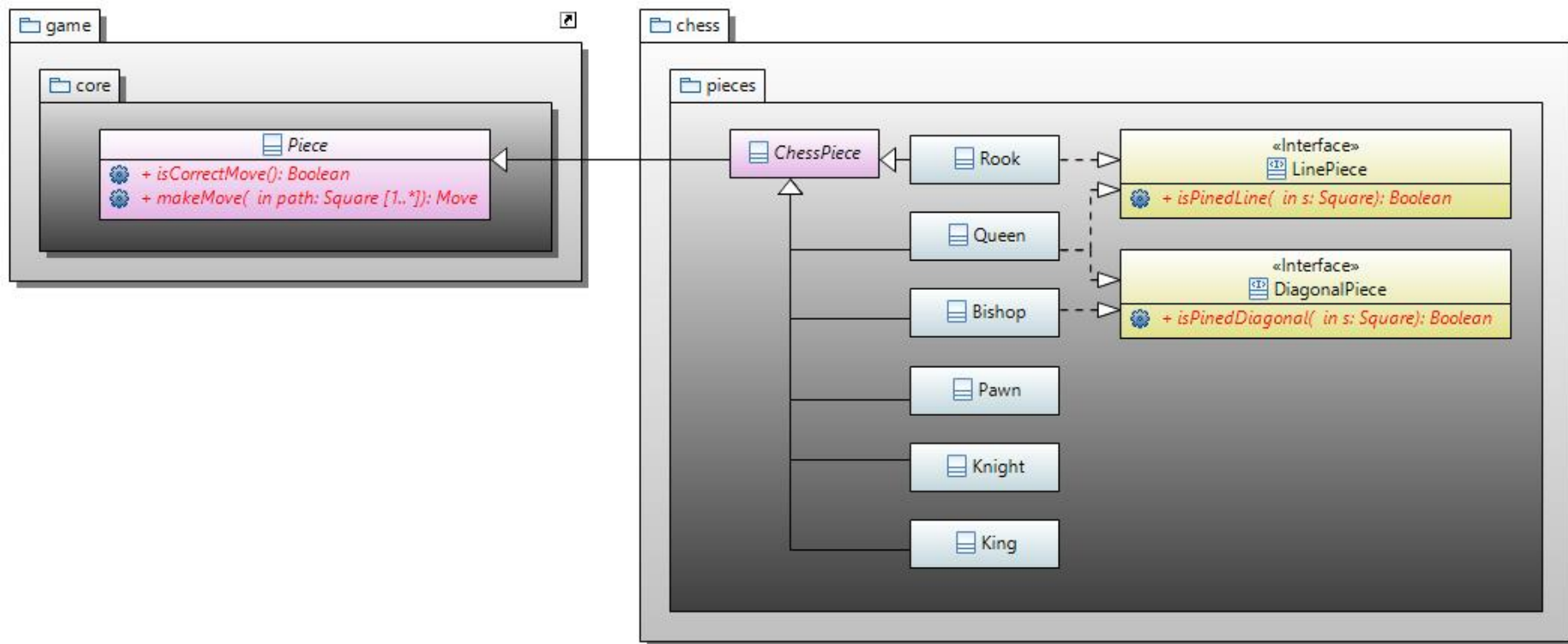


Проектирование классов

Классы-фигуры в шахматной программе

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

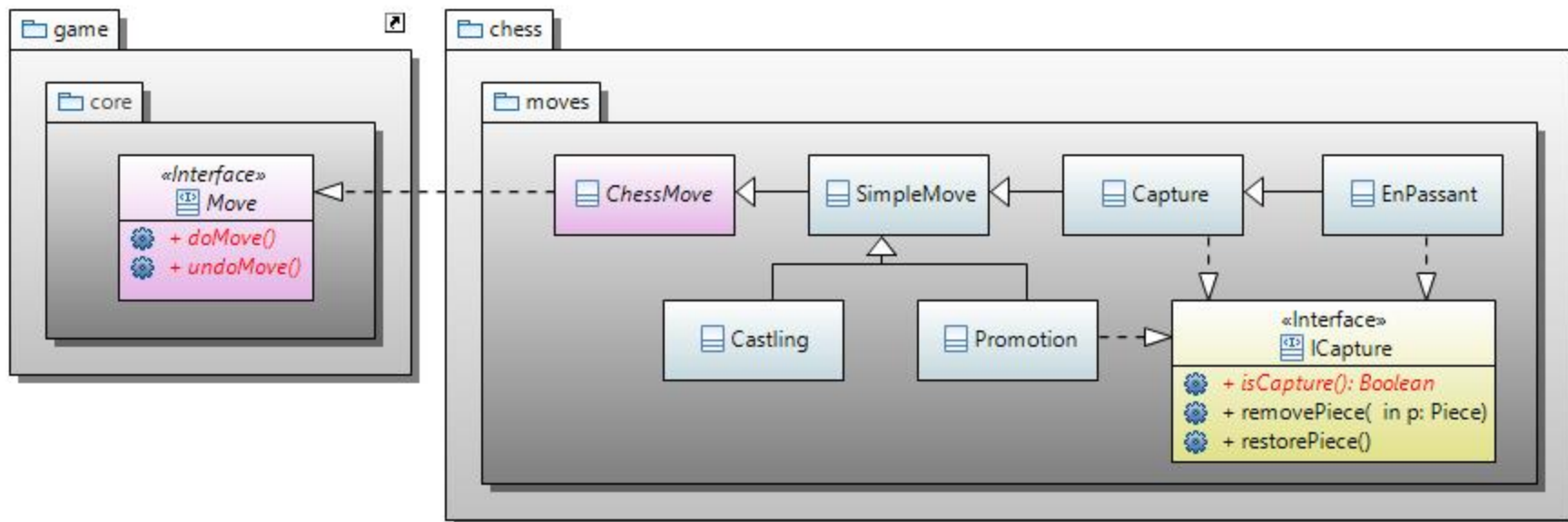


Проектирование классов

Классы-ходы в шахматной программе

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

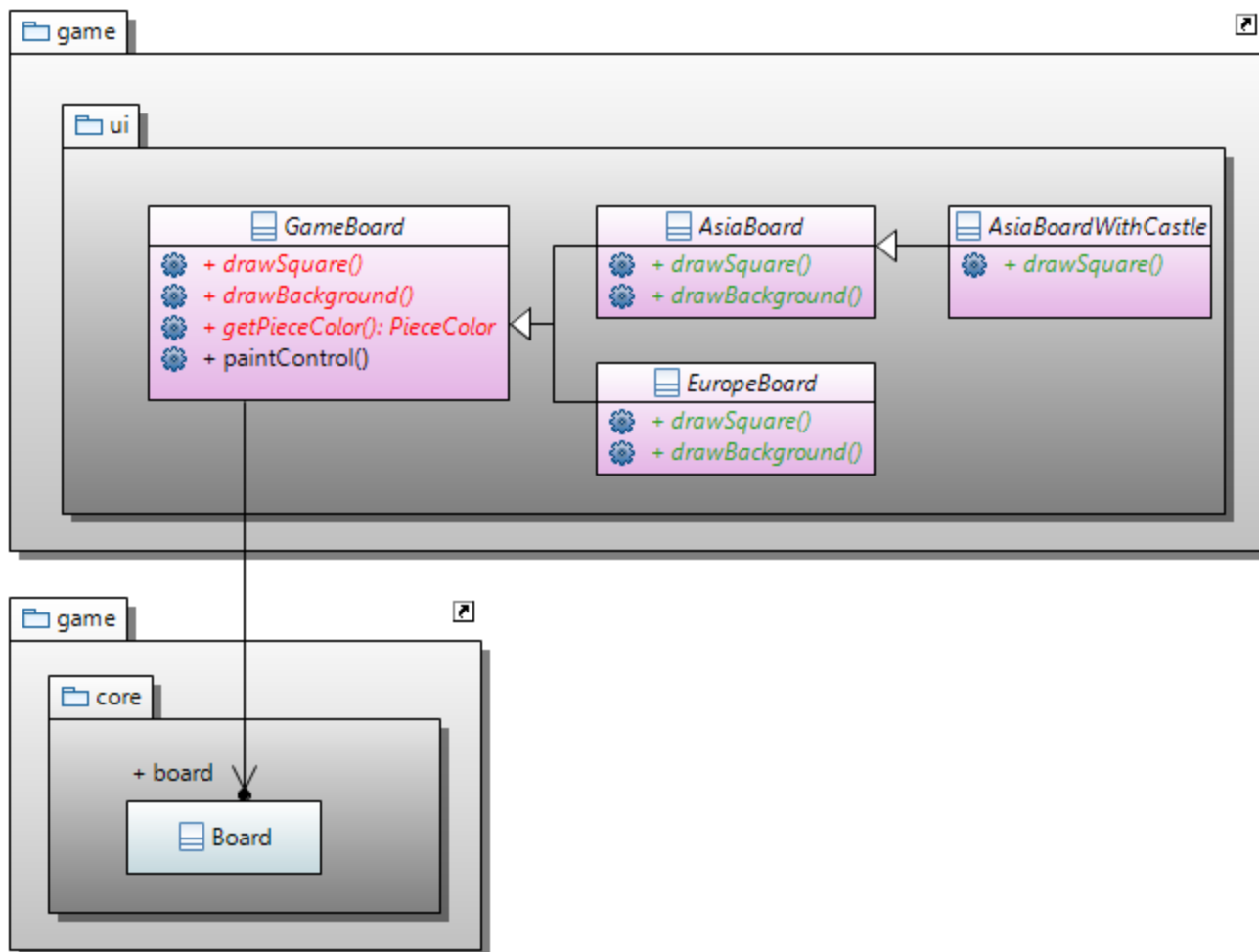


Проектирование классов

Базовые классы интерфейса пользователя

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Объявление абстрактных методов в абстрактном классе

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class GameBoard extends Canvas implements PaintListener {  
    // ...  
    protected Board board;  
  
    public GameBoard(Composite parent, Board board) {  
        super(parent, SWT.NONE);  
  
        this.board = board;  
  
        addPaintListener(this);  
    }  
    // ...  
}
```

Объявление абстрактных методов в абстрактном классе

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class GameBoard extends Canvas implements PaintListener {  
    // ...  
    abstract  
    protected void drawBackground (GC gc, Rectangle area);  
  
    abstract  
    public Image getPieceImage (Piece piece);  
  
    abstract  
    public void drawSquare (GC gc, int v, int h);  
    // ...  
}
```

Использование абстрактных методов в абстрактном классе

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class GameBoard extends Canvas implements PaintListener {
// ...
@Override
public void paintControl (PaintEvent e) {
    GC gc = e.gc;
    Rectangle clientArea = getClientArea();

    drawBackground(gc, clientArea);

    int squareWidth = getClientArea().width / board.nV;
    int squareHeight = getClientArea().height / board.nH;

    for (int v = 0; v < board.nV; v++) {
        for (int h = 0; h < board.nH; h++) {
            drawSquare(gc, v, h);

            drawPiece(gc, v, h);
        }
    }
}
// ...
```

Использование абстрактных методов в абстрактном классе

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class GameBoard extends Canvas implements PaintListener {
// ...
    private void drawPiece (GC gc, int v, int h, int squareWidth, int squareHeight) {
        Piece piece = board.getSquare(v, h).getPiece();
        if (piece == null) return;

        int squareWidth = getClientArea().width / board.nV;
        int squareHeight = getClientArea().height / board.nH;

        int dx = squareWidth / 8;
        int dy = squareHeight / 8;
        int x = v * squareWidth + dx;
        int y = h * squareHeight + dy;

        Image image = getPieceImage(piece);
        Rectangle bounds = image.getBounds();
        gc.drawImage(image,
            0, 0, bounds.width, bounds.height,
            x, y, squareWidth - 2 * dx, squareHeight - 2 * dy);
    }
// ...
}
```

Класс *EuropeBoard* (1)

```
abstract public class EuropeBoard extends GameBoard implements PaintListener {
    private static final Color BLACK = new Color(null, 0, 0, 0);
    private static final Color WHITE = new Color(null, 255, 255, 255);
    private static final Color GREEN = new Color(null, 0, 192, 0);

    public EuropeBoard(Composite parent, Board board) {
        super(parent, board);
    }

    @Override
    public void drawSquare(GC gc, int v, int h, int sw, int sh) {
        int sx = v * sw;
        int sy = h * sh;

        boolean isWhiteSquare = ((v + h) % 2 == 0);
        Color squareColor = isWhiteSquare ? WHITE : GREEN;

        gc.setBackground(squareColor);
        gc.fillRect(sx, sy, sw, sh);

        gc.setForeground(BLACK);
        gc.drawRect(sx, sy, sw, sh);
    }
    // ...
}
```

Класс *EuropeBoard* (2)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
// ...  
  
@Override  
protected void drawBackground(GC gc, Rectangle area) {  
    gc.setForeground(BLACK);  
    gc.drawRectangle(area);  
}  
}
```

Класс AsiaBoard (1)

```
abstract public class AsiaBoard extends GameBoard {
    private static final Color YELLOW = new Color(null, 255, 255, 0);

    public AsiaBoard(Composite parent, Board board) {
        super(parent, board);
    }

    @Override
    public void drawSquare(GC gc, int v, int h) {
        int sw = getClientArea().width / board.nV;
        int sh = getClientArea().height / board.nH;

        int x = v * sw + sw/2;
        int y = h * sh + sh/2;

        if (v != 0) gc.drawLine(x, y, x - sw/2, y);
        if (v != nV-1) gc.drawLine(x, y, x + sw/2, y);

        if (h != 0) gc.drawLine(x, y, x, y - sh/2);
        if (h != nH-1) gc.drawLine(x, y, x, y + sh/2);
    }
    // ...
}
```

Класс AsiaBoard (2)

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

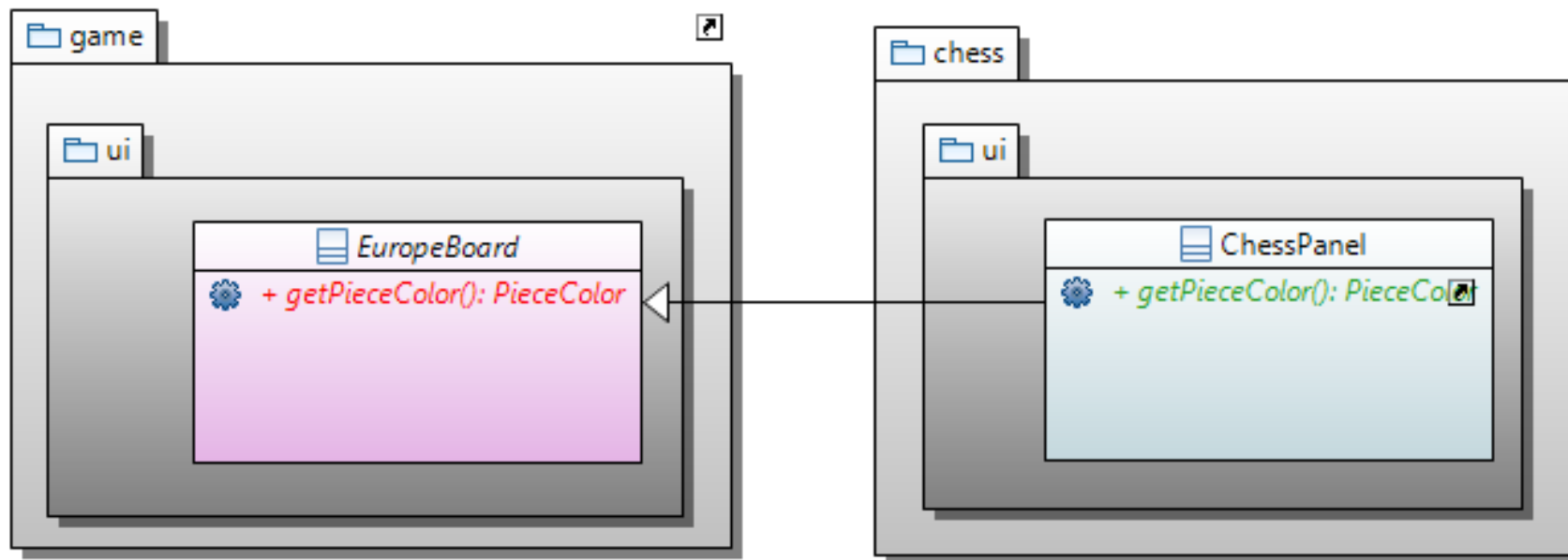
```
// ...  
  
@Override  
protected void drawBackground(GC gc, Rectangle area) {  
    gc.setBackground(YELLOW);  
    gc.fillRect(area);  
}  
}
```


Проектирование классов

Классы интерфейса пользователя шахмат

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Реализация некоторых абстрактных методов в абстрактном классе **EuropeBoard**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class EuropeBoard extends GameBoard {
    public EuropeBoard (Composite composite, Board board) {
        super(composite, board);
    }

    @Override
    protected void drawBackground (GC gc, Rectangle area) {
        // Рисуем фон – деревянную доску.
        Image image = GameImages.woodMedium;
        Rectangle bounds = image.getBounds();

        gc.drawImage(image,
            0, 0, bounds .width, bounds.height,
            area.x, area.y, area.width, area.height);
    }
}
```

Реализация некоторых абстрактных методов в абстрактном классе *EuropeBoard*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class EuropeBoard extends GameBoard {

    @Override
    public void drawSquare (GC gc, int v, int h, int squareWidth, int squareHeight) {
        boolean isWhiteSquare = ((v + h) % 2 == 0);

        Rectangle bounds = GamelImages.woodDark.getBounds();

        if ( !isWhiteSquare )
            gc.drawImage(GamelImages.imageWoodDark,
                0, 0, bounds.width, bounds.height,
                v * squareWidth, h * squareHeight,
                squareWidth, squareHeight);

        gc.setForeground( new Color(null, 0, 0, 0) );
        gc.drawRect(v * squareWidth, h * squareHeight, squareWidth, squareHeight);
    }
}
```

Реализация некоторых абстрактных методов в абстрактном классе **AsiaBoard**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class AsiaBoard extends GameBoard {

    @Override
    protected void drawBackground (GC gc, Rectangle area) {
        // Рисуем фон – деревянную доску.
        Rectangle bounds = GamelImages.ImageWoodLight.getBounds();

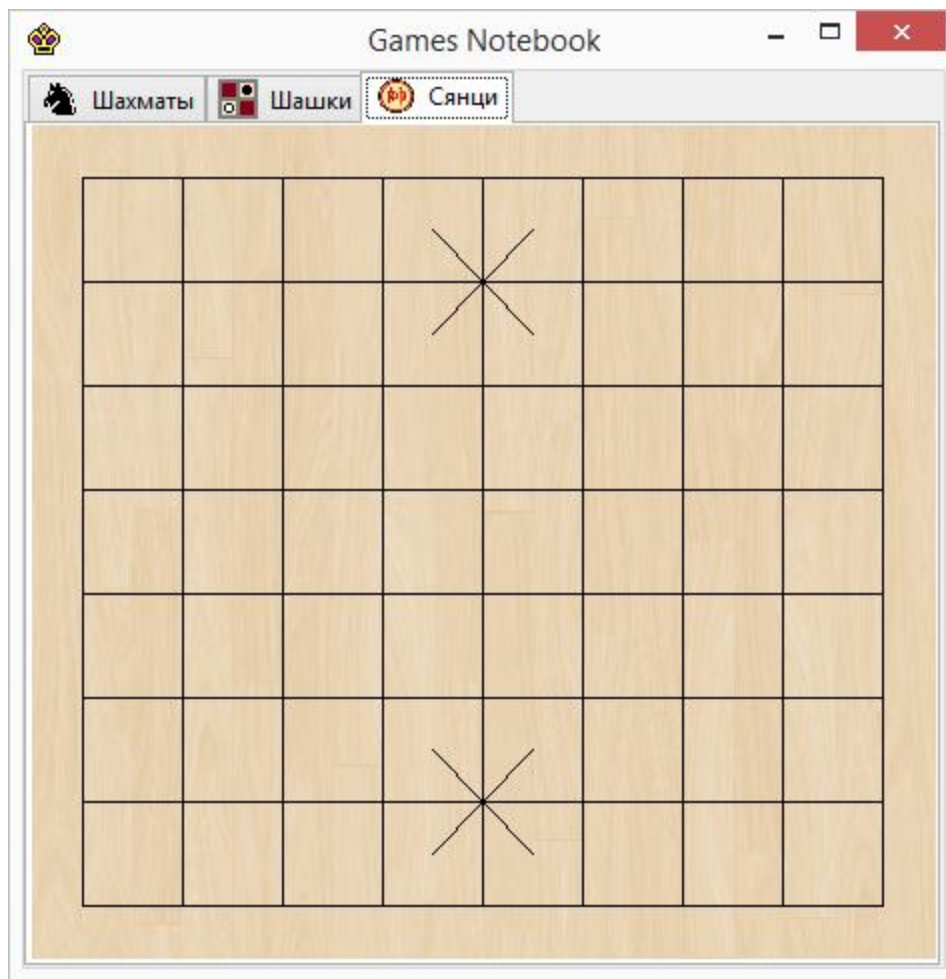
        gc.drawImage(GamelImages.woodLight,
                     0, 0, bounds .width, bounds.height,
                     area.x, area.y, area.width, area.height);
    }

    @Override
    public void drawSquare (GC gc, int v, int h, int squareWidth, int squareHeight)
    {
        // Рисуем клетку – пересечение линий в центре клетки
        // ...
    }
}
```

Блокнот игр. Доска для игры в азиатские игры *Рендзю, Го, Сянци (китайские шахматы)*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Реализация некоторых абстрактных методов в классе EuropeanBoard

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
/**
 * Двухцветная деревянная доска.
 */
public abstract class EuropeanBoard extends AbstractBoardPanel {
    @Override
    protected void drawBoard (GC gc, Rectangle clientArea) {}

    @Override
    protected void drawSquare (GC gc, int v, int h, int sw, int sh) {
        boolean isWhite = ((v + h) % 2 == 0);

        Image squareImage = isWhite
            ? Gamelimages.imageWoodMedium
            : Gamelimages.imageWoodDark;

        gc.drawImage(squareImage, 0, 0, sw, sh, v*sw, h*sh, sw, sh);

        gc.setForeground(COLOR_BLACK);
        gc.drawRect(v * sw, h * sh, sw, sh);
    }
}
```

Реализация
абстрактного метода

Класс *ChessBoard*. Размещение фигур (1)

```
/**
 * Доска с расположенными на ней шахматными фигурами.
 */
public class ChessBoardPanel extends EuropeBoard {
    public ChessBoardPanel(Composite parent) {
        super(parent, 8, 8);

        Chess.putPieces (board, PieceColor.WHITE);
        Chess.putPieces (board, PieceColor.BLACK);
    }

    @Override
    public Image getPieceImage (Piece piece) {
        return getChessPieceImage(piece);
    }

    //...
```

Класс *ChessBoard*. Размещение фигур

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public class Chess extends Game {  
    static public void putPieces (Board board, PieceColor color) {  
        int nV = board.getWidth();  
        int nH = board.getHeight();  
  
        int hPawns = (color == PieceColor.WHITE ? nH-2 : 1);  
        int hPiece = (color == PieceColor.WHITE ? nH-1 : 0);  
  
        for (int v = 0; v < nV; v++)  
            new Pawn(board.getSquare(v, hPawns), color);  
  
        new Rook(board.getSquare(0, hPiece), color);  
        new Knight(board.getSquare(1, hPiece), color);  
        new Bishop(board.getSquare(2, hPiece), color);  
        new Queen(board.getSquare(3, hPiece), color);  
        new King(board.getSquare(4, hPiece), color);  
        new Bishop(board.getSquare(5, hPiece), color);  
        new Knight(board.getSquare(6, hPiece), color);  
        new Rook(board.getSquare(7, hPiece), color);  
    }  
}
```



Расположение
пешек

Класс *ChessBoardPanel*.

Получение изображения фигур

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
/**
 * Доска с расположенными на ней шахматными фигурами.
 */
public class ChessBoardPanel extends EuropeBoard {

    private static Map<Class<? extends Piece>, Image> whites;

    private static Map<Class<? extends Piece>, Image> blacks;

    private static Map<PieceColor, Map<Class<? extends Piece>, Image> > piecelimages;

    static {
        whites = new HashMap<>();
        blacks = new HashMap<>();

        piecelimages = new HashMap<>();
        piecelimages.put(PieceColor.WHITE, whites);
        piecelimages.put(PieceColor.BLACK, blacks);

        // ...
    }
}
```

Класс *ChessBoardPanel*.

Получение изображения фигур

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
// Инициализируем карту изображений белых фигур.  
//  
whites.put(Pawn.class,    ChessImages.imagePawnWhite);  
whites.put(Rook.class,    ChessImages.imageRookWhite);  
whites.put(Knight.class,  ChessImages.imageKnightWhite);  
whites.put(Bishop.class,  ChessImages.imageBishopWhite);  
whites.put(Queen.class,   ChessImages.imageQueenWhite);  
whites.put(King.class,    ChessImages.imageKingWhite);  
  
// Инициализируем карту изображений черных фигур.  
//  
blacks.put(Pawn.class,    ChessImages.imagePawnBlack);  
blacks.put(Rook.class,    ChessImages.imageRookBlack);  
blacks.put(Knight.class,  ChessImages.imageKnightBlack);  
blacks.put(Bishop.class,  ChessImages.imageBishopBlack);  
blacks.put(Queen.class,   ChessImages.imageQueenBlack);  
blacks.put(King.class,    ChessImages.imageKingBlack);
```

Класс *ChessBoardPanel*.

Получение изображения фигур

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
// ...  
  
public Image getPieceImage (Piece piece) {  
    return pieceImages  
        .get( piece.getColor() )  
        .get( piece.getClass() );  
}  
  
} // class ChessBoardPanel
```

Блокнот игр. Шахматные фигуры

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Класс *Piece* – базовый класс для всех клеточных игр.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
abstract public class Piece {  
    public PieceColor color = PieceColor.WHITE;  
  
    protected Square square;  
  
    public Piece(PieceColor color) {  
        this.color = color;  
    }  
  
    /**  
     * Допустим ли ход фигурой на заданную клетку.  
     */  
    abstract public boolean isCorrectMove (Square s);  
  
    /**  
     * Передвинуть фигуру на заданную клетку.  
     */  
    abstract public Move moveTo (Square s);
```

Класс *ChessPiece* – базовый класс для всех клеточных игр.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
/**
 * Базовый класс для всех шахматных фигур.
 */
public abstract class ChessPiece extends Piece {
    public ChessPiece(PieceColor color) {
        super(color);
    }

    /**
     * Общее свойство шахматных фигур - нельзя бить фигуру своего цвета.
     */
    @Override
    public boolean isCorrectMove (Square s) {
        if (s.isEmpty()) return true;

        // Фигуры разного цвета?
        return s.getPiece().color != color;
    }
}
```

Класс *Pawn* - пешка.

Реализация абстрактных методов

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

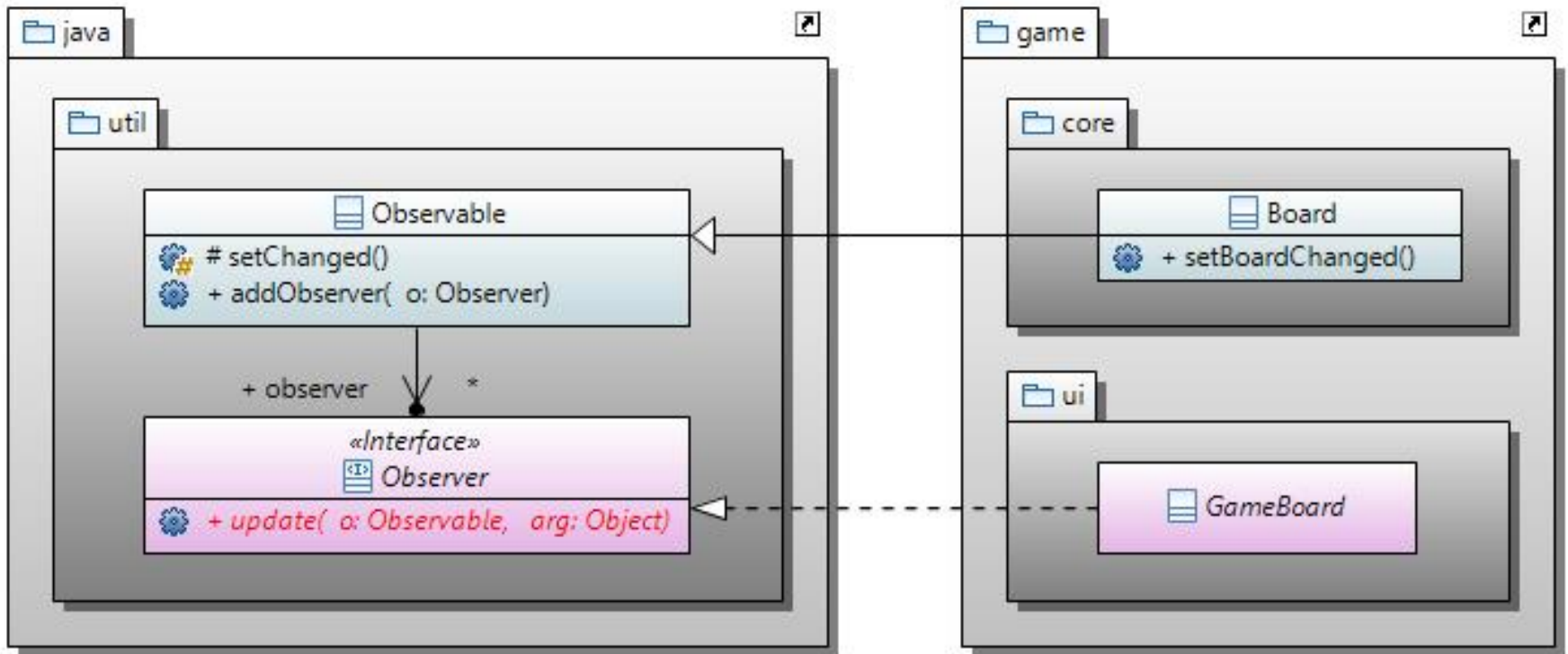
```
public class Pawn extends ChessPiece {  
    public Pawn(PieceColor color) {  
        super(color);  
    }  
    @Override  
    public boolean isCorrectMove (Square target) {  
        // ...  
    }  
  
    @Override  
    public Move moveTo (Square target) {  
        // ...  
    }  
  
    @Override  
    public String toString() { return ""; }  
}
```

Класс *Observable* **и** **интерфейс *Observer***

Использование класса *Observable* и интерфейса *Observer*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Обозреваемый класс Board.

```
public class Board extends Observable {
```

```
// ...
```

```
public void setBoardChanged ( ) {  
    super.setChanged();  
}
```

```
// ...
```

```
}
```

Можно добавлять
обозревателей

Изменения произошли.
Уведомляем
обозревателей

Класс-обозреватель **GameBoard**

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class GameBoard extends Canvas
    implements PaintListener, MouseListener, Observer
{
    public GameBoard(Composite parent, Board board) {
        super(parent, SWT.NONE);

        this.board = board;
        addPaintListener(this);

        addMouseListener( this );
        board.addObserver( this );
        board.setBoardChanged();
    }

    @Override
    public void update (Observable o, Object arg) {
        update();
        redraw();
    }
}
```

Класс **GameBoard**
будет слушать
изменения
на **доске**

Класс **GameBoard**
добавил себя
как слушателя изменения
на **доске**

Действия при изменении
на **доске**

Программирование взаимодействия с пользователем

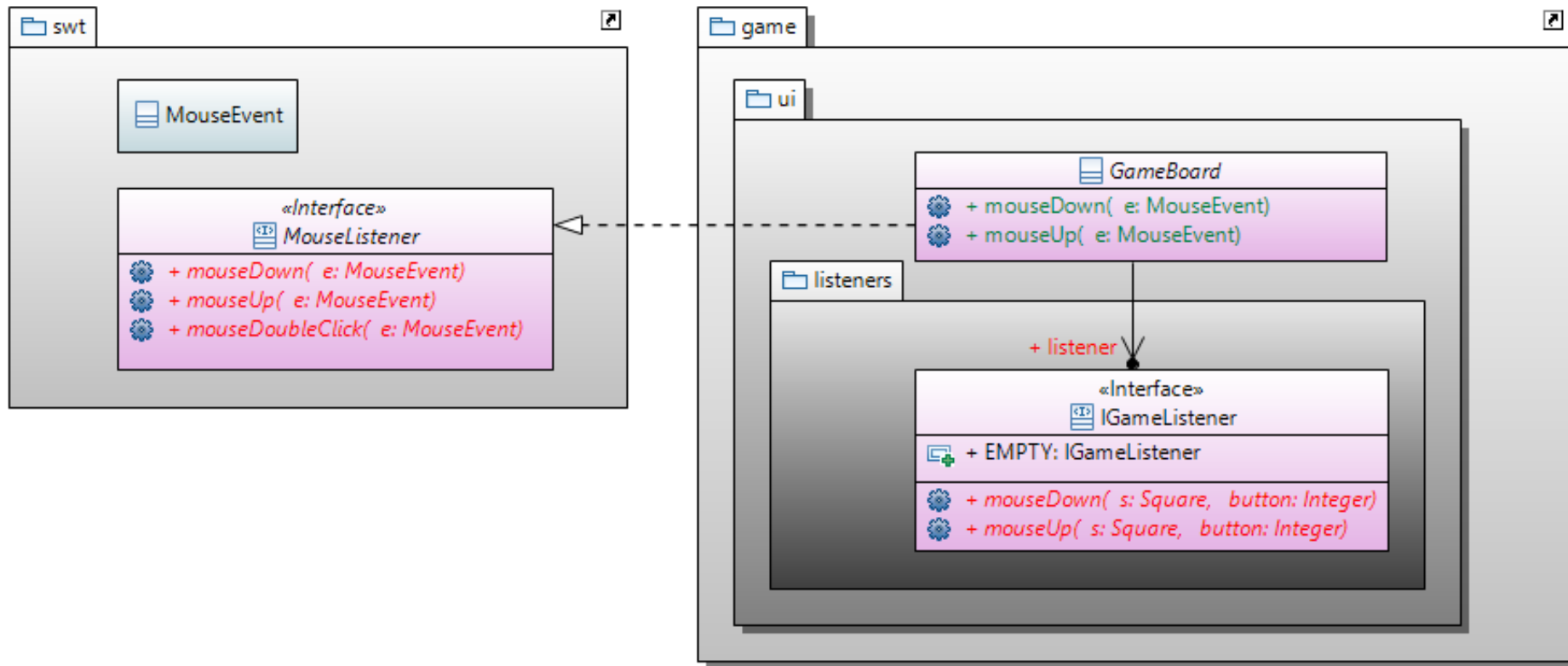
События от клавиатуры мыши

Использование класса *Observable*

и интерфейса *Observer*

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



SWT. Программирование клавиш мыши.

Перемещение фигуры на доске

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class GameBoard extends Canvas  
    implements PaintListener, MouseListener, Observer
```

```
{  
    protected Board board;
```

Класс **GameBoard** будет
слушать клавиши мыши

```
    public GameBoard(Composite parent, Board board) {  
        super(parent, SWT.NONE);
```

```
        this.board = board;
```

```
        addPaintListener( this );  
        addMouseListener( this );
```

```
    }
```

Класс **GameBoard**
добавил себя
как слушателя клавиш
мыши

Блокнот игр. Реакция на события от МЫШИ

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class GameBoard extends Canvas
    implements PaintListener, MouseListener, Observer {
// ...

    protected IGameListner listener = IGameListner.EMPTY;

    @Override
    public void mouseDown(MouseEvent e) {
        Square s = getSquare(e);
        listener.mouseDown(s, e.button);
    }

    @Override
    public void mouseUp(MouseEvent e) {
        Square s = getSquare(e);
        listener.mouseUp(s, e.button);
    }

    @Override
    public void mouseDoubleClick(MouseEvent e) {}
// ...
```


Блокнот игр. Реакция на события от МЫШИ

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public abstract class GameBoard extends Canvas
    implements MouseListener, Observer {
// ...
protected Square getSquare(MouseEvent e) {
    int sw = getSquareWidth();
    int sh = getSquareWidth();
    int v = e.x / sw;
    int h = e.y / sh;

    return board.getSquare(v, h);
}

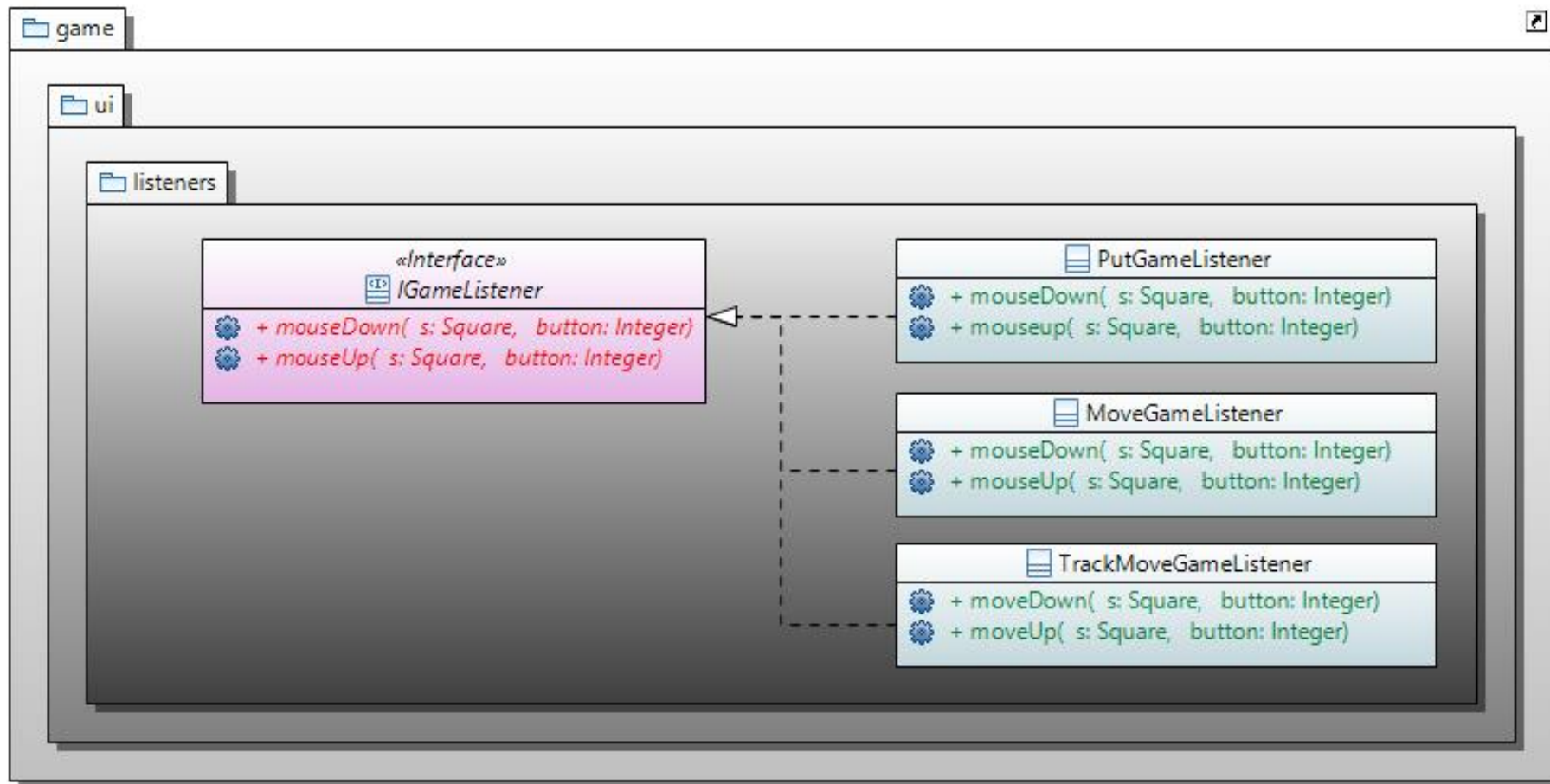
private int getSquareHight() { return getClientArea().height / board.nH; }

private int getSquareWidth() { return getClientArea().width / board.nV; }
// ...
```

Слушатели ходов на доске

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Класс *MovePieceListener*.

Фигура перемещается на одно поле

```
abstract
public class MovePieceListener implements IGameListner {
    private Piece selectedPiece;

    private Square selectedSquare;

    private Cursor savedCursor;

    private PieceColor moveColor = PieceColor.WHITE;

    private Board board;

    private GameBoard panel;

    public MovePieceListener (GameBoard panel) {
        this.board = panel.board;
        this.panel = panel;
    }

    // ...
```

Класс *MovePieceListener*.

Установка фигуры на новое поле

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
@Override  
public void mouseDown(Square mouseSquare, int button) {  
    if ( mouseSquare.isEmpty())  
        return;  
  
    selectedPiece = mouseSquare.getPiece();  
    if (selectedPiece.getColor() != moveColor)  
        return;  
  
    savedCursor = panel.getCursor();  
  
    selectedSquare = mouseSquare;  
    selectedSquare.removePiece();  
  
    panel.pieceToCursor(selectedPiece);  
  
    board.setBoardChanged();  
    panel.redraw();  
}
```

Класс *GameBoard*.

Изображение фигуры в изображение курсора

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
/**
 * Сделать изображение фигуры изображением курсора.
 *
 * @param piece фигура изображение которой "перемещается" в курсор.
 */
public void pieceToCursor (Piece piece) {
    Image image = getPieceImage(piece);
    imageToCursor(image);
}
```

Класс *GameBoard*.

Изображение фигуры в изображение курсора

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
Cursor boardCursor
    = new Cursor(Display.getCurrent(), SWT.CURSOR_ARROW);

public void imageToCursor(Image image) {
    int sw = getClientArea().width / board.nV;
    int sh = getClientArea().height / board.nH;

    int pw = sw - sw/8; // Ширина фигуры в клетке.
    int ph = sh - sh/8; // Высота фигуры в клетке.

    ImageData imageData = image.getImageData().scaledTo(pw,ph);

    boardCursor.dispose();

    Display display = Display.getCurrent();

    boardCursor = new Cursor(display, imageData, sw/2, sh/2);
    setCursor(boardCursor);
}
```

Класс *MovePieceListener*.

Выбор фигуры для перемещения

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

@Override

```
public void mouseUp(Square mouseSquare, int button) {  
    if (selectedSquare == null) return;  
  
    mouseSquare.setPiece(selectedPiece);  
  
    if (selectedPiece.isCorrectMove (mouseSquare) ) {  
        Move move = selectedPiece.moveTo (mouseSquare);  
        move.doMove();  
        board.history.addMove(move);  
        moveColor = selectedPiece.getOpponentColor();  
    }  
  
    selectedPiece = null; selectedSquare = null;  
  
    panel.setCursor(savedCursor);  
  
    board.setBoardChanged();  
    panel.redraw();  
}
```

Класс *MovePieceListener*.

Выбор фигуры для перемещения

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

@Override

```
public void mouseUp(Square mouseSquare, int button) {  
    if (selectedSquare == null)  
        return;  
  
    selectedPiece.setSquare(mouseSquare);  
    //mouseSquare.setPiece(selectedPiece);  
    if (selectedPiece.isCorrectMove(mouseSquare)) {  
        Move move = selectedPiece.moveTo(mouseSquare);  
        //move.doMove();  
        //board.history.addMove(move);  
        moveColor = selectedPiece.getOponentColor();  
    }  
    selectedPiece = null;  
    selectedSquare = null;  
    panel.setCursor(savedCursor);  
    board.setBoardChanged();  
    panel.redraw();  
}
```


Использование класса *MovePieceListener* для перемещения фигур игры Шахматы

МГУ им. М.В.Ломоносова. Факультет ВМК.

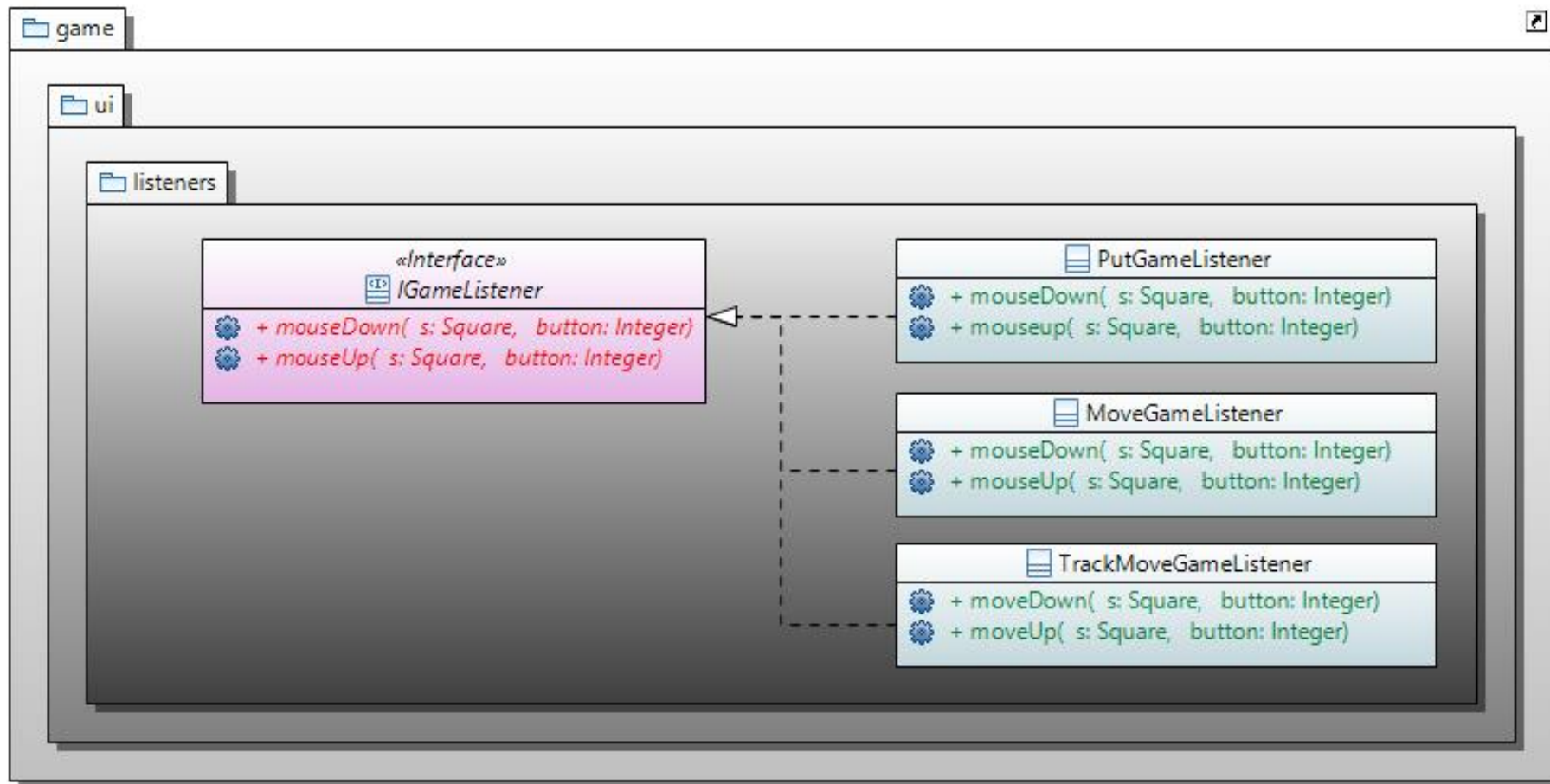
Романов Владимир Юрьевич ©2025

```
public class ChessBoardPanel extends EuropeBoard {  
  
    // ...  
  
    public ChessBoardPanel (Composite composite) {  
        super(parent, new Board(8, 8));  
  
        Chess.putPieces(board);  
  
        listener = new MovePieceListener(this);  
    }  
  
    // ...  
}
```

Слушатели ходов на доске

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025



Класс *PutPieceListener*.

Фигуры игры ставятся на доску

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

abstract

```
public class PutPieceListener implements IGameListner {
```

```
    private PieceColor moveColor = PieceColor.WHITE;
```

```
    private Board board;
```

```
    private GameBoard panel;
```

```
    public PutPieceListener(GameBoard panel) {
        this.board = panel.board;
        this.panel = panel;
    }
```

```
    @Override
```

```
    public void mouseUp(Square s, int button) {}
```

```
// ...
```

Класс *PutPieceListener*.

Фигуры игры ставятся на доску

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

@Override

```
public void mouseDown(Square mouseSquare, int button) {  
    if (! mouseSquare.isEmpty()) return;
```

```
    Piece piece = getPiece (mouseSquare, moveColor);
```

```
    if (! piece.isCorrectMove (mouseSquare)) {  
        piece.remove(); return;  
    }
```

```
    Move move = piece.makeMove (mouseSquare);  
    move.doMove();  
    board.history.addMove(move);
```

```
    moveColor = getOponentColor(moveColor);  
    panel.imageToCursor( getPiecImage(piece, moveColor) );
```

```
    board.setBoardChanged();  
    panel.redraw();
```

```
}
```

```
abstract public Piece getPiece(Square square, PieceColor color);
```

```
//...
```

Использование класса *PutPieceListener* для установки фигур игры Реверси

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
public class ReversiBoardPanel extends GreenBoard {

    public ReversiBoardPanel(Composite composite, int nHoles) {
        super(composite, Reversi.getInitBoard(nHoles) );

        listener = new PutPieceListener (this) {
            @Override
            public Image getPiecelmage (Piece piece, PieceColor color) {
                return ReversiBoardPanel.this.getPiecelmage(piece, color);
            }

            @Override
            public Piece getPiece (Square square, PieceColor color) {
                return new Stone(square, color);
            }
        };
    }

    // ...
}
```

События от перемещения мыши

SWT. Программирование перемещения мыши.

```
abstract
public class GameBoard extends Canvas
implements PaintListener, MouseListener, MouseMoveListener, Observer
{

public GameBoard(Composite parent, Board board) {
    super(parent, SWT.DOUBLE_BUFFERED);

    this.board = board;

    addPaintListener(this);

    addMouseListener(this);
    addMouseMoveListener(this);
```

Класс **GameBoard**
будет слушать
перемещение мыши

Класс **GameBoard**
добавил себя
как слушателя
перемещения мыши

Использование класса *MovePieceListener* для перемещения фигур игры Шахматы

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
/**
 * Слушатель события перемещения мыши.
 */
protected IMouseMoveListener mouseMoveListener
    = new MovePiecePromptListener(this);

@Override
public void mouseMove(MouseEvent e) {
    Square s = getSquare(e);

    if (s != null)
        mouseMoveListener.mouseMove(s);
}
```


События от перемещения мыши

SWT. Программирование перемещения мыши.

```
abstract
public class GameBoard extends Canvas
implements PaintListener, MouseListener, MouseMoveListener, Observer
{

public GameBoard(Composite parent, Board board) {
    super(parent, SWT.DOUBLE_BUFFERED);

    this.board = board;

    addPaintListener(this);

    addMouseListener(this);
    addMouseMoveListener(this);
```

Класс **GameBoard**
будет слушать
перемещение мыши

Класс **GameBoard**
добавил себя
как слушателя
перемещения мыши

События от колеса мыши

SWT. Программирование колеса мыши.

```
// ! Что бы доска получала фокус добавим слушателя клавиатуры.  
// После этого доска начнет получать события от колеса мыши.  
addKeyListener(new KeyListener() {  
    @Override  
    public void keyPressed(KeyEvent e) {}  
  
    @Override  
    public void keyReleased(KeyEvent e) {}  
});
```

SWT. Программирование колеса мыши.

```
// Добавим слушателя колеса мыши.  
addMouseListener (new MouseWheelListener() {  
    @Override  
    public void mouseScrolled (MouseEvent e) {  
        if (e.count > 0)  
            board.history.toPrevMove();  
        else board.history.toNextMove();  
  
        board.setBoardChanged();  
    }  
});
```

Списки библиотеки SWT в интерфейсе пользователя

SWT. Программирование списков в интерфейсе пользователя.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
String [ ] players = { "player1", "player2" };
```

```
List list = new List (group, SWT.BORDER / SWT.SINGLE);  
players.forEach(p -> list.add(p) );
```

```
list.setForeground(BORDER_COLOR);  
list.setBackground(LIST_COLOR);  
list.select(playerNumber);
```

```
list.addListener (SWT.Selection, event ->  
    List list = (List) event.widget;  
    int selection = list.getSelectionIndices() [0];  
);
```

Кнопки библиотеки SWT в интерфейсе пользователя

SWT. Программирование кнопок в интерфейсе пользователя.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
Button startButton = new Button (group, SWT.NONE);  
startButton.setText("Старт");  
startButton.setLayoutData(data);  
  
startButton.addListener (SWT.Selection, event -> {  
    game.initBoardDefault();  
    board.startGame();  
});
```

SWT. Программирование радио-кнопок в интерфейсе пользователя.

МГУ им. М.В.Ломоносова. Факультет ВМК.

Романов Владимир Юрьевич ©2025

```
Group group = new Group (this, SWT.SHADOW_IN);  
group.setText("Доска");
```

```
// ...
```

```
Button b = new Button (group, SWT.RADIO);  
b.setText( "text1");  
b.setSelection(true);
```

```
b.addListener(SWT.Selection, event -> {  
    if ( b.getSelection() )  
        gamePanel.resizeBoard(nV, nH);  
});
```

```
// ...
```