

Skillbox

ОБРАЗОВАТЕЛЬНЫЙ ОНЛАЙН-КУРС

# Программист 1С-Битрикс

**Модуль 1. Философия разработчика,  
подготовка рабочего места**



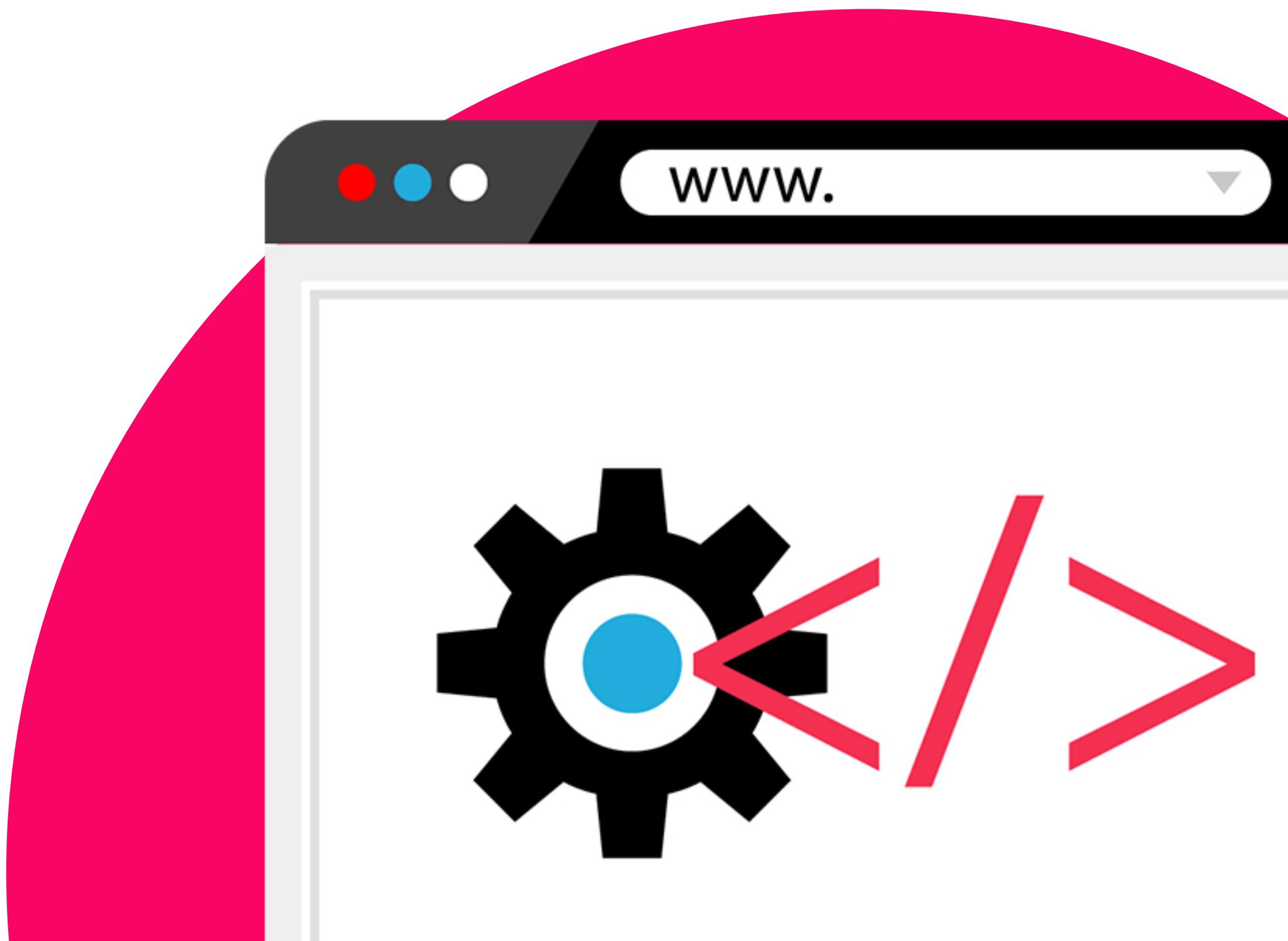
Преподаватель курса

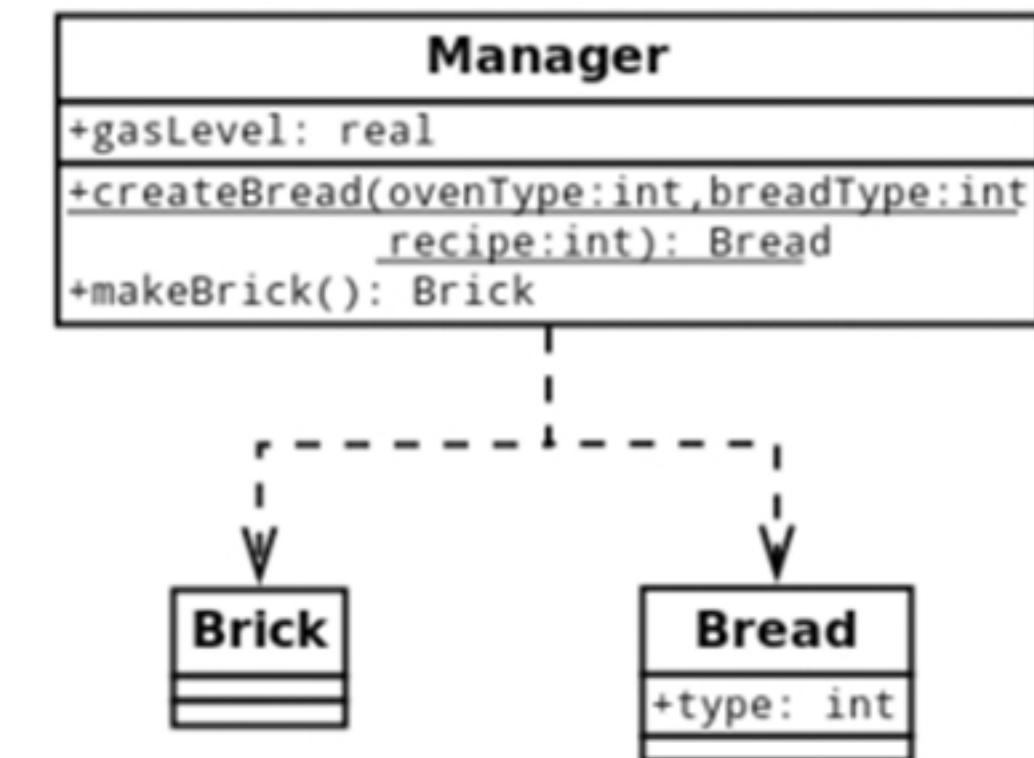
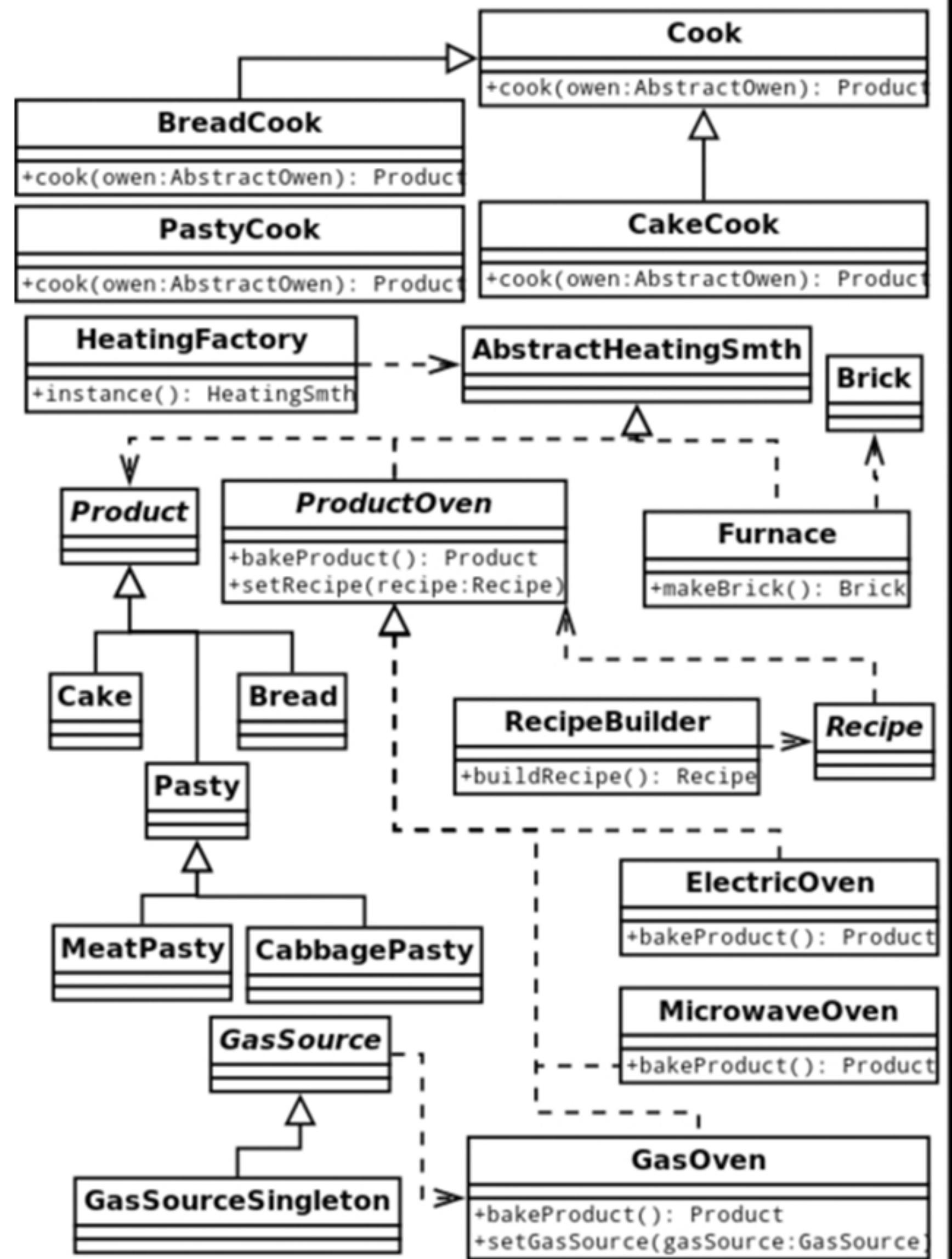
## **Вячеслав Фирсунин**

- Окончил МАТИ по специальности Кибернетика, после чего занялся веб-разработкой
- За семь лет успешно запустил более 20 крупных проектов. Два крайних проекта - онлайн франчайзинг Позитроника ([positronica.ru](http://positronica.ru)), сайт фабрики Яршинторг ([kolesatyt.ru](http://kolesatyt.ru))
- На данный момент Вячеслав является руководителем веб-разработки компании Ашманов и партнеры

# Разрабатывать хорошо или быстро?

- Про баланс в разработке  
<https://habrahabr.ru/post/153225/>





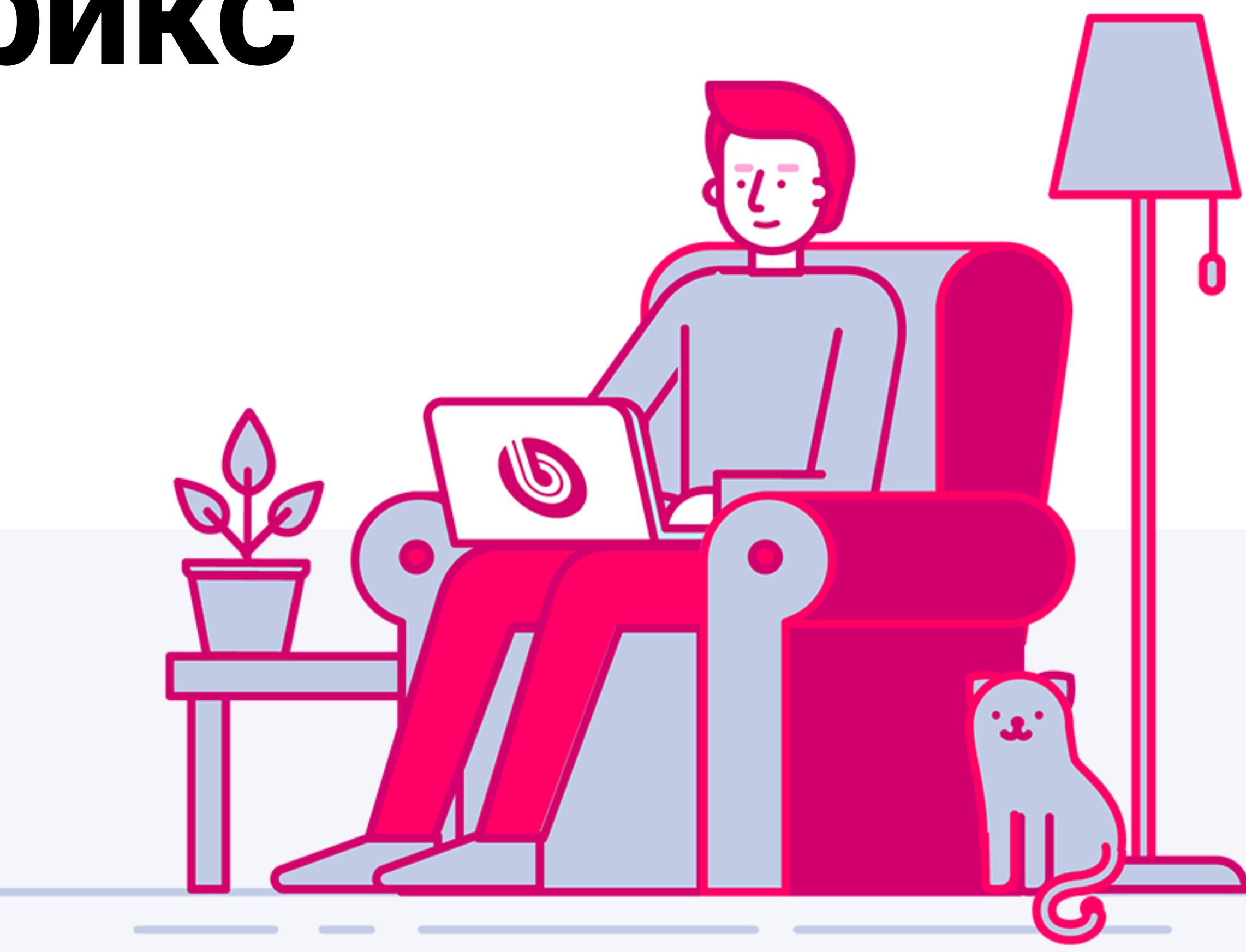
# Выбор архитектуры исходя из целей и задач

# Доверяй, но проверяй! Два реальных примера

- Работа аjax компонентов
- Хранение картинок



# Регистрация на хостинге и установка Битрикс



# Архитектура Битрикс

- Ядро
- Файлы
- База

# В какой программе работать и в чем разница?

 **PhpStorm**



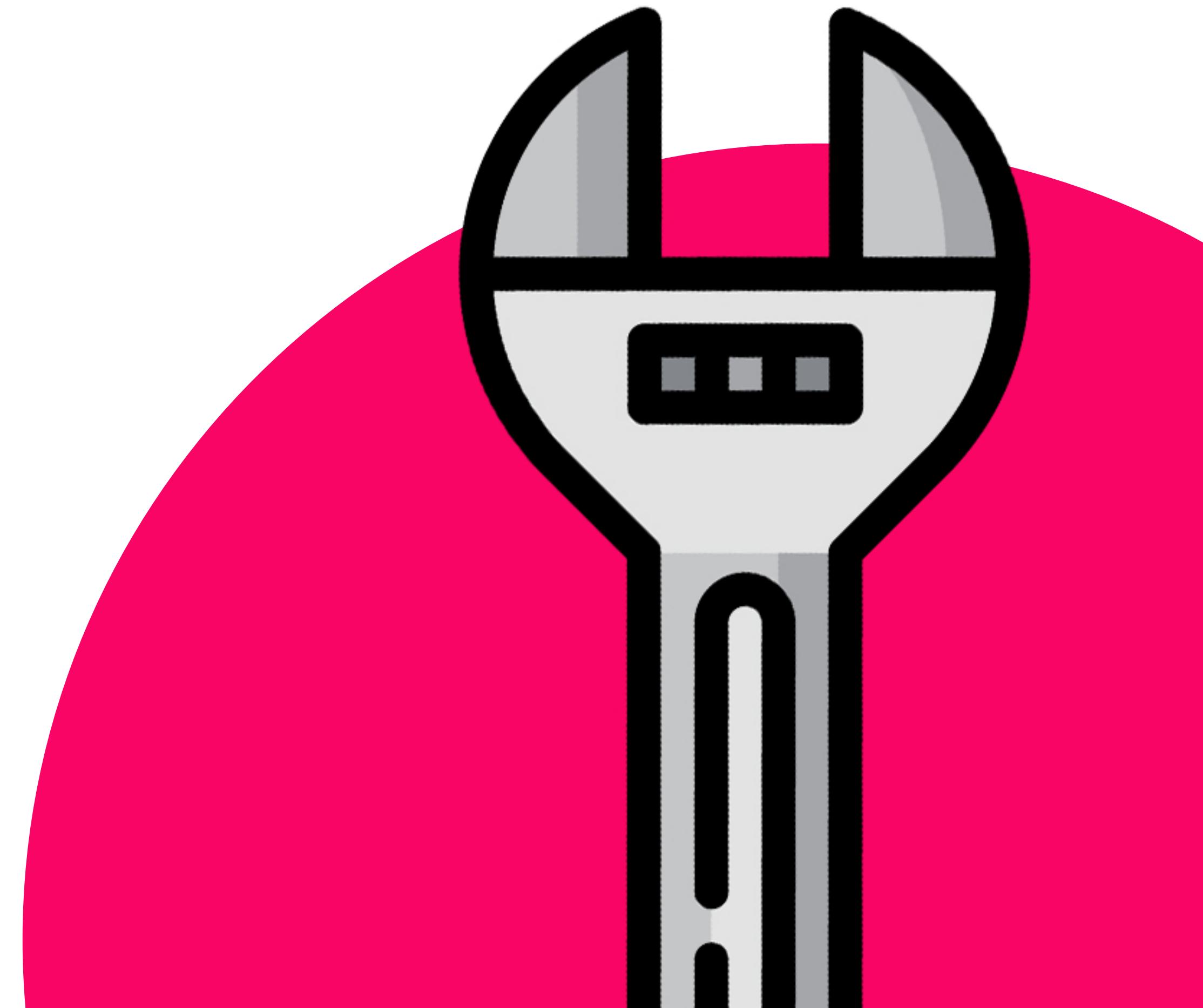
 **eclipse**

 **NetBeans**

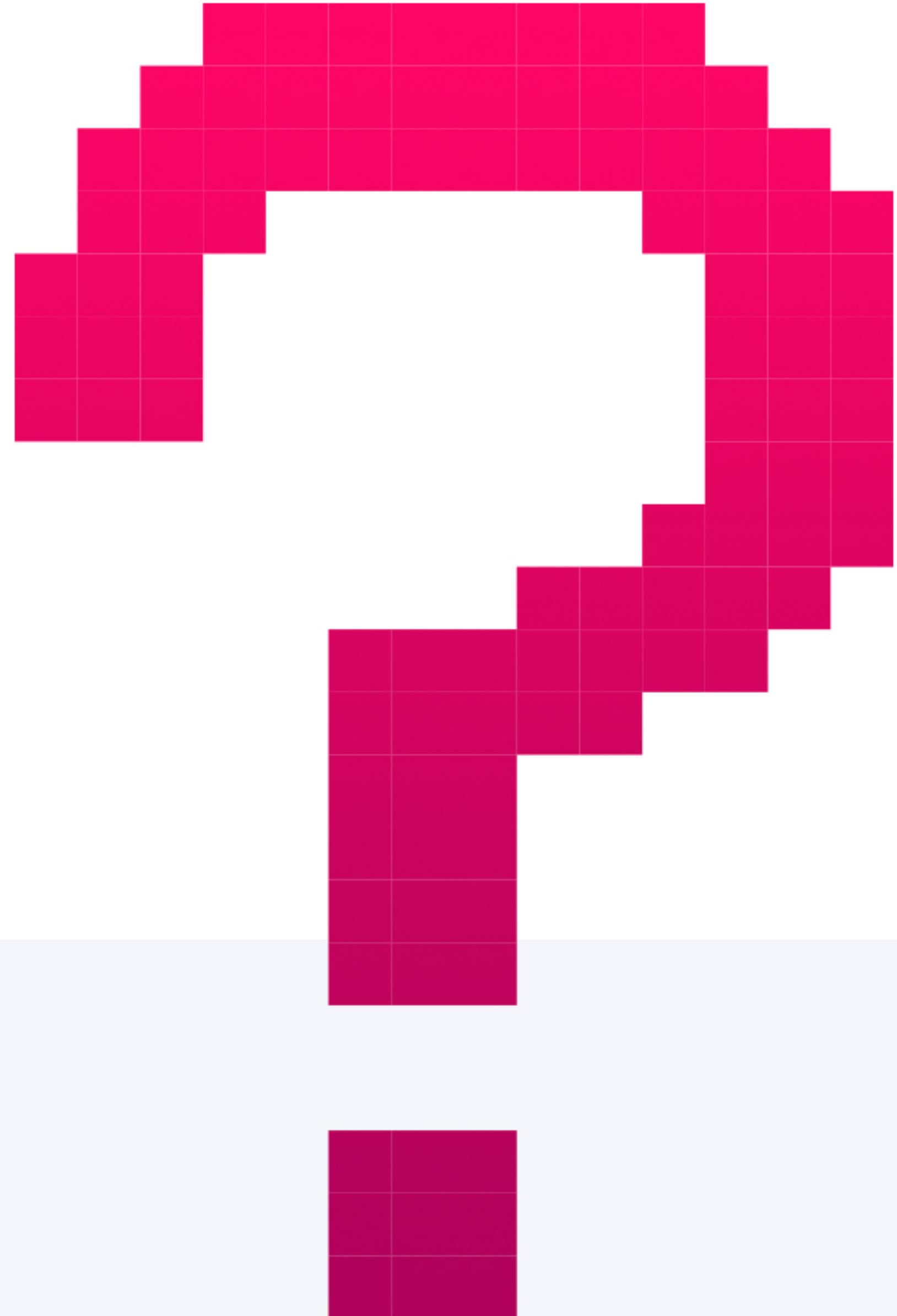
 **Sublime Text**

# Настройка PhpStorm

- Создание проекта
- Настройка удаленной площадки
- Горячие клавиши
- Сниппеты



# **Что такое Git и зачем**



# Пример ведения веток



# **Что добавляем в гит конкретно в битриксе**



# Работа в команде



# Домашнее задание

- Установите и настройте PHP Storm согласно пройденным шагам.

В форму домашнего задания

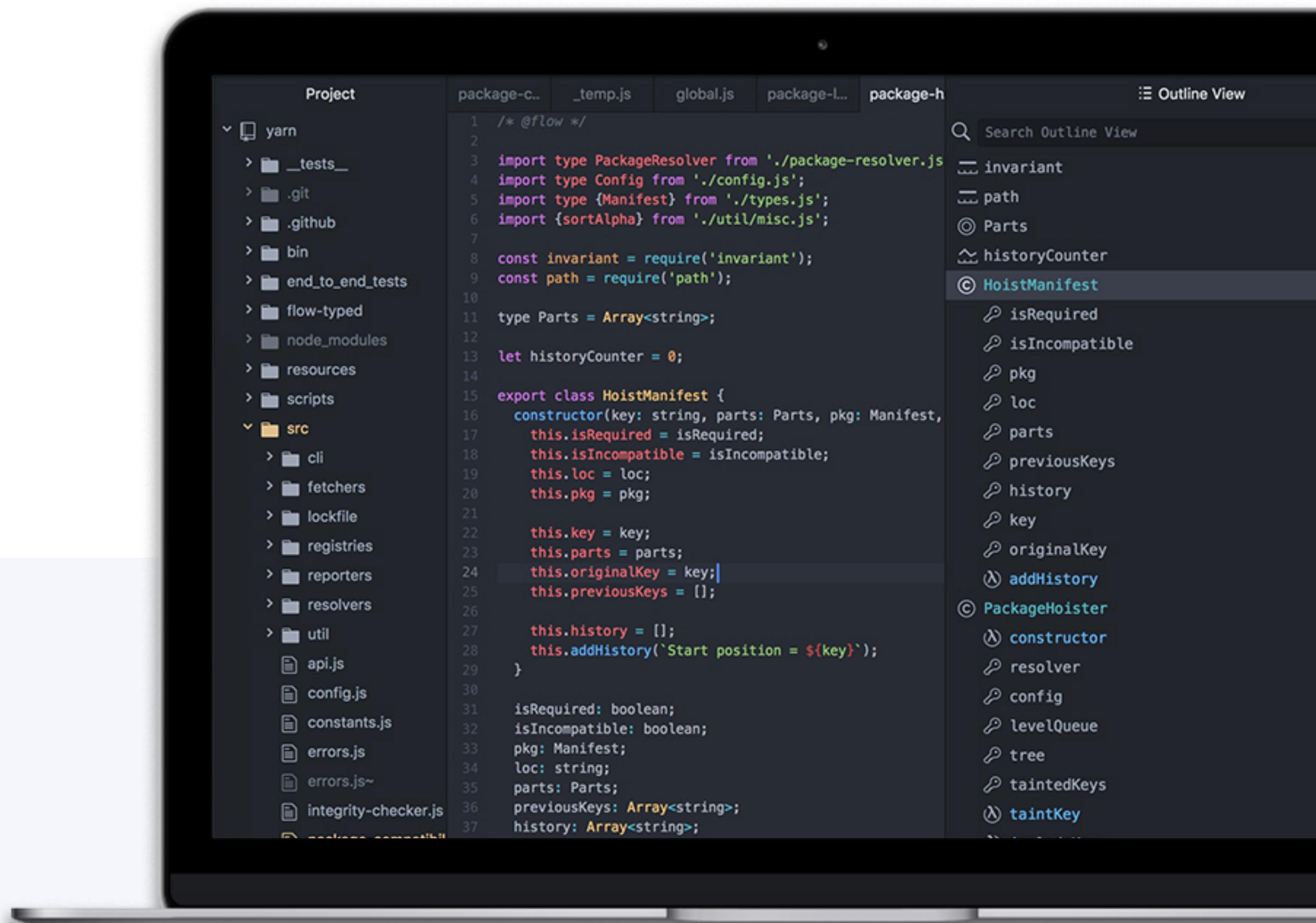
вышлите скрин установленной IDE

- Зарегистрируйтесь на хостинге

- Разверните первоначальный проект битрикс с демо данными.

В форму домашнего задания

пришлите ссылку на проект.



The screenshot shows the PhpStorm IDE interface. On the left, the Project tool window displays a file structure under the 'yarn' root, including folders like '\_tests\_', '.git', '.github', 'bin', 'end\_to\_end\_tests', 'flow-typed', 'node\_modules', 'resources', 'scripts', and 'src'. The 'src' folder contains subfolders 'cli', 'fetchers', 'lockfile', 'registries', 'reporters', 'resolvers', 'util', and files 'api.js', 'config.js', 'constants.js', 'errors.js', 'errors.js~', and 'integrity-checker.js'. On the right, the Outline View tool window is open, showing a hierarchical tree of symbols from a file named 'HoistManifest'. The tree includes nodes for 'invariant', 'path', 'Parts', 'historyCounter', 'HoistManifest' (selected), 'isRequired', 'isIncompatible', 'pkg', 'loc', 'parts', 'previousKeys', 'history', 'key', 'originalKey', 'addHistory', 'PackageHoister', 'constructor', 'resolver', 'config', 'levelQueue', 'tree', 'taintedKeys', and 'taintKey'. The code editor at the bottom shows the source code for the 'HoistManifest' class.

```
/* @flow */
import type PackageResolver from './package-resolver.js';
import type Config from './config.js';
import type Manifest from './types.js';
import {sortAlpha} from './util/misc.js';

const invariant = require('invariant');
const path = require('path');

type Parts = Array<string>;
let historyCounter = 0;

export class HoistManifest {
    constructor(key: string, parts: Parts, pkg: Manifest, loc: string) {
        this.isRequired = isRequired;
        this.isIncompatible = isIncompatible;
        this.loc = loc;
        this.pkg = pkg;
        this.key = key;
        this.parts = parts;
        this.originalKey = key;
        this.previousKeys = [];
    }

    addHistory(`Start position = ${key}`);
}

isRequired: boolean;
isIncompatible: boolean;
pkg: Manifest;
loc: string;
parts: Parts;
previousKeys: Array<string>;
history: Array<string>;
```