Среда разработки RubyMine

Этот урок про IDE (по-русски, среда разработки): установим и научимся пользоваться средой разработки **RubyMine**, поймём чем она отличается от текстового редактора **Sublime**.

### План урока

1. Что такое IDE (среда разработки)?
2. Установка RubyMine
3. Настройка RubyMine
4. Разбираемся с русскими символами и кодировками
5. Создаем первый проект в RubyMine

Что такое IDE?

Если подойти к вопросу формально, то IDE — это [Integrated Development Environment](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8), или интегрированная среда разработки. Другими словами, это рабочее место разработчика, внутри которого у него под рукой будут все необходимые ему инструменты: редактор файлов, консоль, просмотр папок проекта и т. д.

### Основные функции IDE:

Во-первых, это текстовый редактор. Вы уже знаете про Sublime, он тоже очень удобный, красивый и функциональный. В любой IDE есть подсветка кода, а также проверка его на ошибки «на лету».

Во-вторых, в IDE, как правило, встроена консоль: вам не придётся переключаться между окнами, теряя на этом время. Файлы сохраняются автоматически, а запуск программы — нажатие одной клавиши. Плюс , конечно же, горячие клавиши, чтобы всё получалось ещё быстрее.

В-третьих, вы прямо в IDE видите структуру вашей программы. Без этого просто никуда, если программа у вас сложная. Вам не нужно переключаться в менеджер файлов (проводник в Windows) и искать нужный файл. Весь ваш проект постоянно у вас под рукой.

В-четвёртых, в течение этого курса мы расскажем вам о таких инструментах как отладчик, тесты и система контроля версий. Все эти инструменты (а также многие другие) можно встроить в IDE, о которой мы вам сегодня расскажем.

### Установка RubyMine

Давайте, наконец, поставим себе IDE для разработки на Ruby — **RubyMine**. Для установки **RubyMine** на компьютер с Windows вам понадобится установленный **ruby-installer** и умение пользоваться **консолью**.

IDE бывают разные. Выбор IDE чаще всего зависит от того языка, на котором вы работаете. Для Ruby есть несколько вариантов, которые относительно популярны: [NetBeans](https://ru.wikipedia.org/wiki/NetBeans), [Komodo](https://en.wikipedia.org/wiki/Komodo_Edit) и [RubyMine](https://ru.wikipedia.org/wiki/RubyMine), на которой мы сегодня и сосредоточимся.

Мы выбрали RubyMine во-первых, потому что он максимально заточен под разработку на ruby. Во-вторых, потому что он основан на среде разработки [IntelliJ IDEA](https://ru.wikipedia.org/wiki/IntelliJ_IDEA), которая является самой удобной IDE из всех существующих на данный момент. И в-третьих, её разработали наши, питерские ребята из [JetBrains](https://www.jetbrains.com/).

В процессе обучения мы будем постепенно раскрывать интересные детали и полезные хитрости этой среды разработки.

Пробная версия **RubyMine** длится 30 дней (чего вам вполне хватит для того, чтобы пройти курс). А если решите купить эту систему — это будет не самым плохим вложением денег.

**ВАЖНО**! Все задания и программы курса работают и без RubyMine. Так что наличие этой среды у вас желательно, но не обязательно.

Для установки RubyMine её необходимо скачать с официального сайта:

<https://www.jetbrains.com/ruby/>

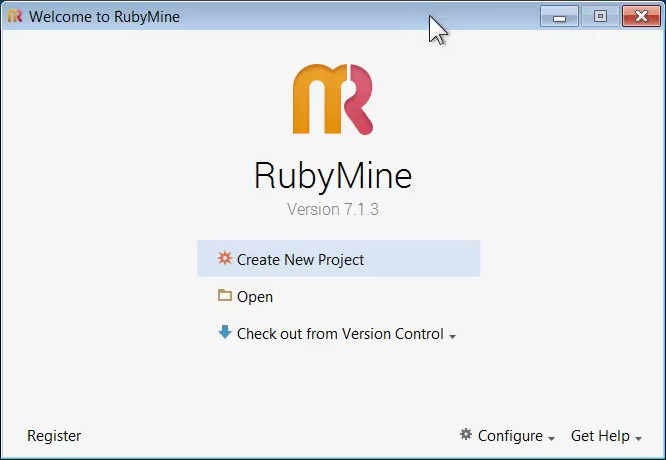
В RubyMine можно работать под любой операционной системой. Сайт сам определит, какая у вас операционная система и предложит скачать именно нужный дистрибутив.

После этого, как обычно устанавливаем программу:



### Что такое проект?

После установки RubyMine предложит вам создать проект:



Для удобства в IDE принято выделять любую программу в проект. Проект логически объединяет все файлы одной программы и позволяет удобно их просматривать и открывать.

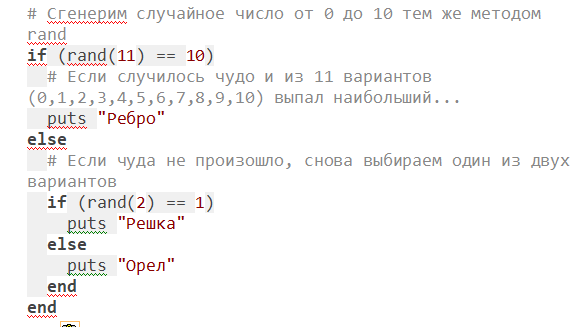
Грубо говоря, это папка на вашем компьютере, где лежат все файлы и подпапки вашей программы. Пока вы учитесь, у вас будет много учебных проектов и все они будут простыми (чаще всего, состоять из одного-двух файлов), но когда дело доходит до серьёзных программ, количество файлов одной программы может измеряться сотнями и даже тысячами. Поэтому концепция проектов себя полностью оправдывает.

### Проект «Монетка»

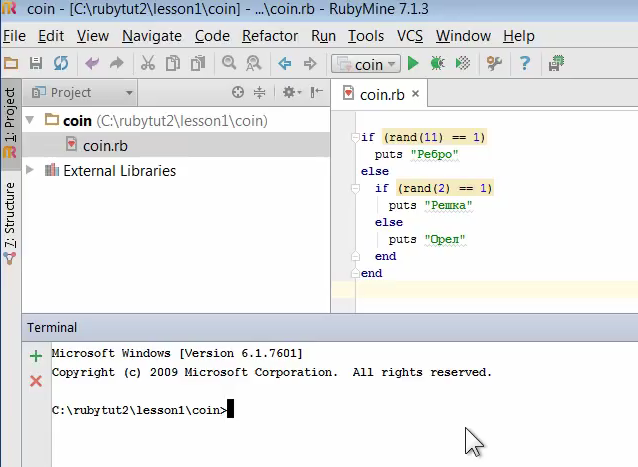
Давайте создадим наш первый проект. Для начала создайте на диске новую папку rubytut2, в которой мы будем хранить все наши проекты для второго курса. В этой папке создайте папку lesson1. А затем создайте в RubyMine новый проект. Для этого выберите в меню запуска Create New Project и укажите путь к новому проекту:

C:\rubytut2\lesson1\coin

После создания нового проекта в папке lesson1 появится папка coin. Скопируйте туда файлы из вашего домашнего задания к 4-му уроку базового блока или возьмите их из материалов к этому уроку.



Посмотрите, теперь в RubyMine видно всю структуру вашей программы. Она, конечно, не сложная. Но всё равно удобно. Нажмите на основном файле вашей программы правой кнопкой мыши и запустите его, выбрав пункт Run.

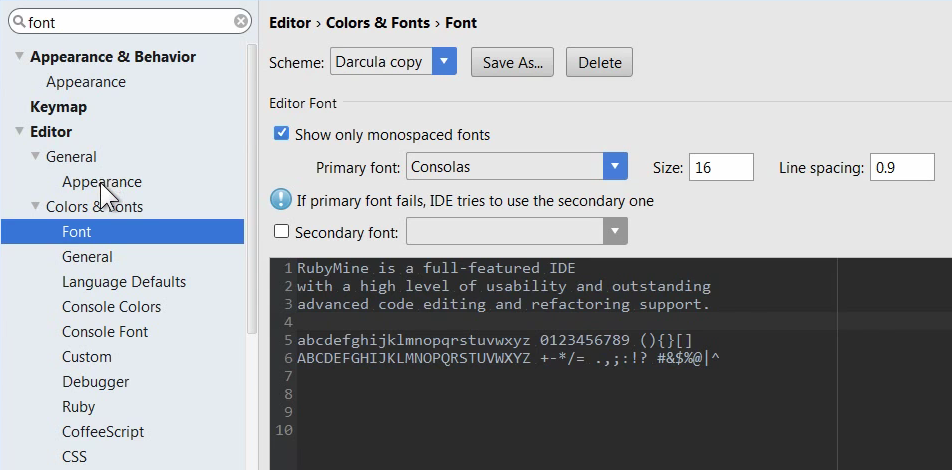


Вы увидите консоль RubyMine.

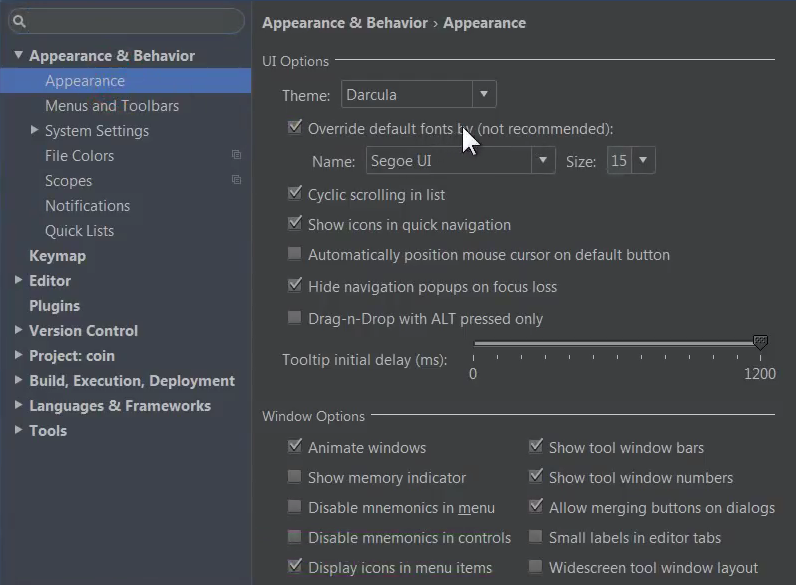
### Настройка RubyMine

Давайте немного настроим RubyMine, чтобы нам было поудобнее работать. Для этого можно либо выбрать в меню File → Settings, либо нажать Ctrl + Alt + S.

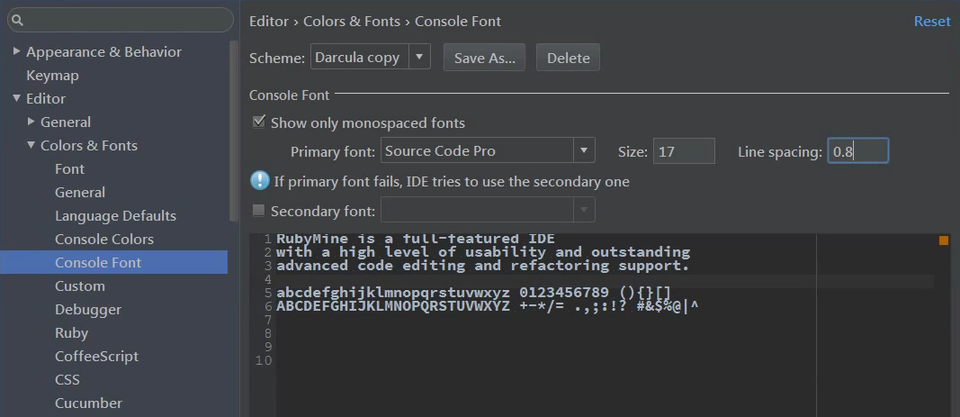
Найдите настройки шрифтов, вбив в поисковой строке слово font. Вы увидите что-то подобное:



Загляните также во вкладку Appearance:



А также настройте внешний вид консоли:



И наконец, самое важное. Чтобы наши программы работали с русским языком в RubyMine, нужно настроить кодировку. Но для начала давайте, наконец, разберёмся с тем, что это такое.

### Что такое кодировки?

Компьютер, на самом деле, не понимает букв и других символов. Данные в компьютере записываются в виде чисел. Поэтому чтобы научить его работать с буквами, придумали кодировки. Кодировка — это соответствие последовательности нулей и единиц (битов, которые собираются в байты и более сложные конструкции) символам на экране вашего компьютера. Это как шифровки, которые дети шлют друг другу, играя в разведчиков.

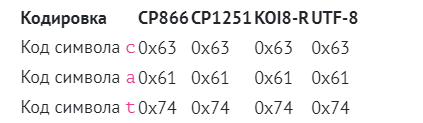
К сожалению, каждая компания, занимающаяся производством компьютеров или программного обеспечения считала своим долгом придумать для себя собственную кодировку, поэтому этих кодировок огромный зоопарк. Этот зоопарк усиливается тем, что кодировки бывают разными в зависимости от того, символы какого языка нам нужно закодировать.

Подробнее про кодировки можно почитать на [Википедии](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D0%B1%D0%BE%D1%80_%D1%81%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB%D0%BE%D0%B2).

А пока, простой пример:

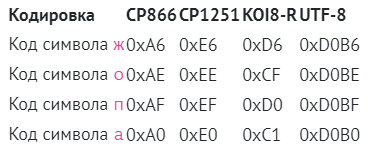
Рассмотрим три кодировки, которые можно использовать для кодирования русских символов: [CP1251](https://ru.wikipedia.org/wiki/Windows-1251) — стандартная кодировка Windows для кириллицы, [CP866](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D1%8C%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BA%D0%BE%D0%B4%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) — кодировка IBM для кириллицы и [KOI8-R](https://en.wikipedia.org/wiki/KOI8-R) — кодировка для русских символов, разработанная в СССР. А также рассмотрим кодировку [UTF-8](https://ru.wikipedia.org/wiki/UTF-8), в которой есть не только русский язык, но и многое другое.

Вот какие коды нужно использовать для того, чтобы закодировать слово cat, набранное, конечно же, латиницей, в каждой из этих кодировок:



Обратили внимание, что символы для всех трёх букв совпадают? То же самое будет и с любой латинской буквой. При разработке каждой из этих кодировок (слава Богу) учли, что для того, чтобы они были совместимы, неплохо бы закодировать латинские символы одинаковыми числами. Нижняя часть вышеперечисленных таблиц кодировки полностью соответствует самой первой кодировке ASCII.

Давайте же посмотрим, что происходит с русскими буквами:



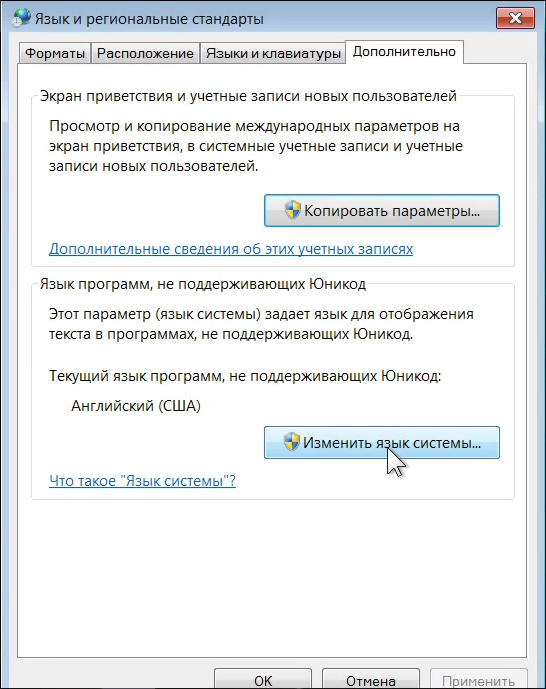
Как видите, всё не так однозначно. Именно поэтому вы часто видите кракозябры на своих компьютерах. Фактически, шифровку, которую вам передают ребята из соседнего двора (с другого сайта или одна программа — другой), ваш компьютер расшифровывает неправильно. Ваш компьютер видит число и предполагает, что кодировка одна (допустим cp1251), а тот, кто это число создавал (другой компьютер или ваша собственная программа), предполагал другую кодировку, например UTF-8. Поэтому при преобразовании чисел в символы происходят ошибки и вместо нужных символов выводятся другие.

Чтобы этот аспект не мешал нам в наших программах, давайте настроим все наши инструменты на работу в одной и той же кодировке: UTF-8. Она разрабатывалась сильно позже всех и включает в себя, помимо русского, множество языков. Если говорить коротко, следует предпочитать кодировку UTF-8 всем другим кодировкам при разработке программ и создании сайтов.

### Настройка кодировки UTF-8 в RubyMine

Для того, чтобы проблема кодировок нас не беспокоила, по крайней мере на протяжении этого курса, давайте сделаем несколько настроек.

Во-первых, настройте кодировку консоли Windows. Для этого в **Панели инструментов** выберите настройку языков. Во вкладке **Дополнительно** нажмите кнопку **Изменить язык системы** и выберите там **Русский (Россия)**. После этого придётся перезагрузить компьютер.



После перезагрузки компьютера откройте командную строку и выполните команду chcp. Если вы всё настроили правильно, то вы увидите надпись

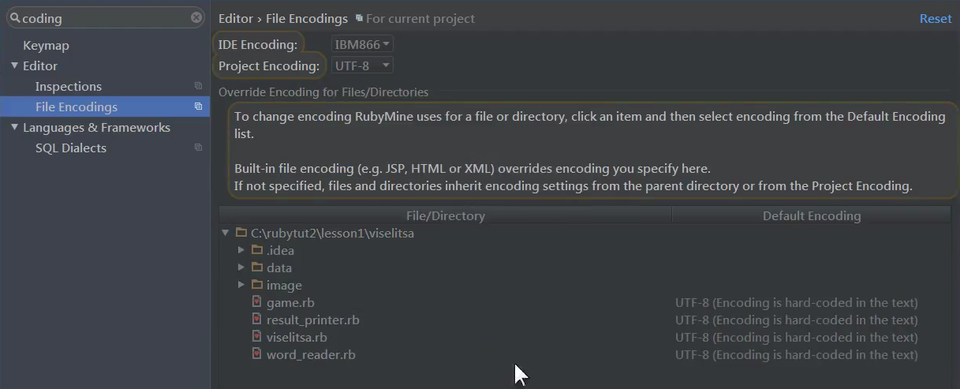
> chcp

Текущая кодовая страница: 866

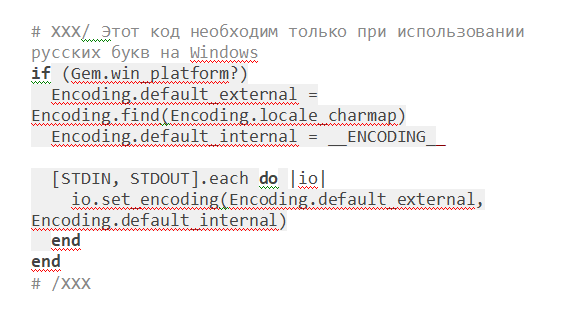
Во-вторых, настроим кодировки в RubyMine. Откройте настройки (**File -> Settings**) и найдите там настройки кодировок (**File Encodings**).

Установите кодировку **UTF-8** для проектов (**Project Encoding**) и для файлов настроек по умолчанию (**Default encoding for properties files**).

Внимание, теперь для самой среды разработки (**IDE Encoding**) выберите кодировку **IBM866**. Это важно, т.к. мы настроили нашу консоль для работы именно с этой кодировкой.



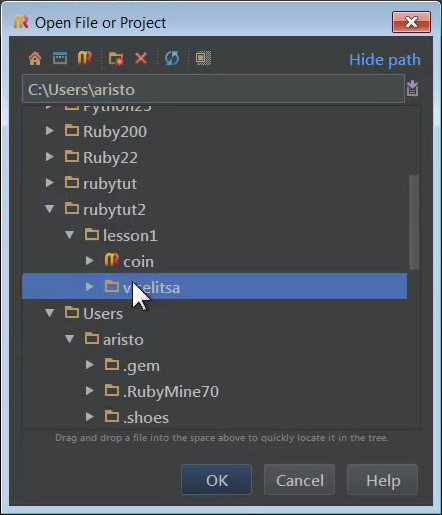
В-третьих, в тексте наших программ нам необходимо будет каждый раз (увы!) вставлять в самом начале вот такой код, если мы собираемся использовать русские буквы (как для вывода информации, так и в случае, если хотим, чтобы пользователь имел возможность вводить русские символы).



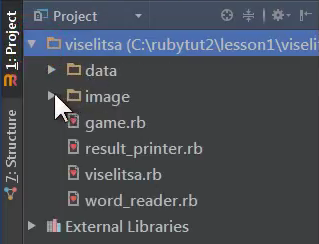
### Экспортируем проект «Виселица»

Теперь, когда мы настроили кодировки, можем экспортировать нашу «Виселицу» и не бояться, что у нас будут проблемы с отображением русских символов.

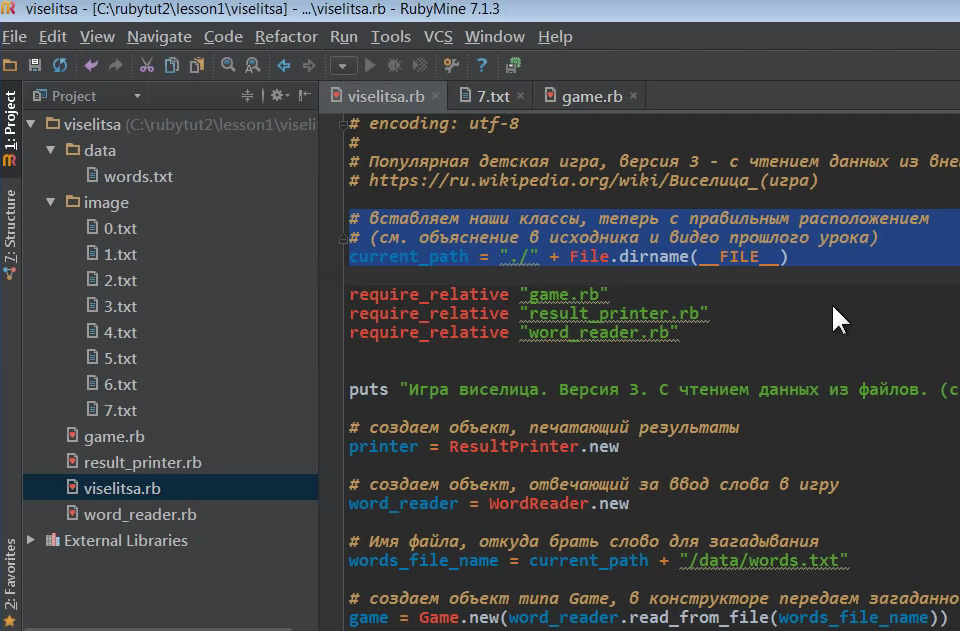
Скопируйте программу viselitsa в нашу папку урока rubytut2/lesson1. В RubyMine выберите **File -> Open** и укажите путь к нашей «Виселице».



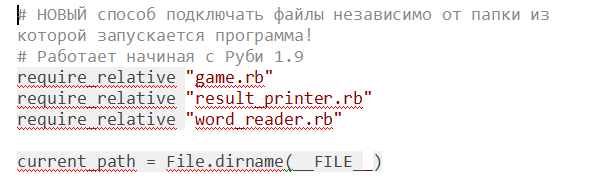
Теперь вы видите структуру проекта: все файлы и папки с данными (data, images):



Откройте файлы в редакторе RubyMine, освойтесь. Посмотрите, как всё круто и удобно.



Важно! Обратите внимание, как мы поменяли в тексте программы подключение файлов в проект с помощью команды require\_ralative. Это необходимо сделать, чтобы программу можно было запускать из RubyMine. Ну и вообще, так подключать файлы более правильно..



Как замените require на require\_relative и уберёте './' из строки current\_path, запустите «Виселицу»: кликните по файлу main.rb в проводнике RubyMine правой кнопкой мыши и выберите Run.