

ЛЕКЦИЯ 27

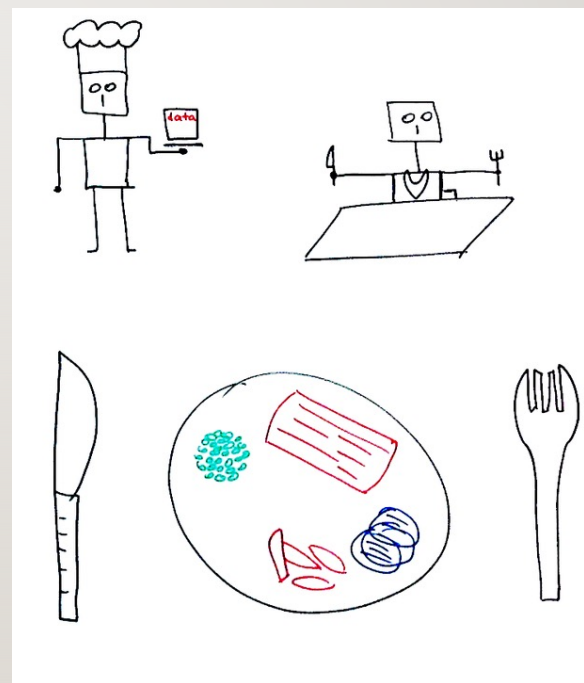
ХРАНЕНИЕ ДАННЫХ, XML, HTML

ПЛАН ЗАНЯТИЯ

1. Как по-человечески хранить данные
 2. Формат данных XML
 3. Чтение XML в Ruby
 4. Немного про HTML
- Этим уроком мы начинаем блок курса, посвященный тому, как в реальном мире хранятся данные. В следующих уроках мы рассмотрим форматы данных XML, JSON и познакомимся с базами данных.

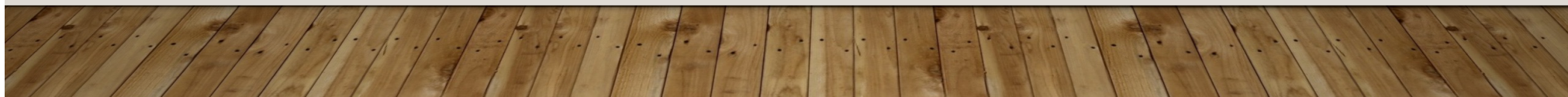
ХРАНЕНИЕ ДАННЫХ ДЛЯ ПРОГРАММ

- Программы работают с информацией, как человек с пищей — создают, готовят, потребляют, перерабатывают, показывают. Для того, чтобы программам (и программистам) было удобнее с ней работать, информацию нужно как-то хранить, в каком-то удобном виде предоставлять пользователям или другими программам.



ХРАНЕНИЕ ДАННЫХ ДЛЯ ПРОГРАММ

- Информация бывает очень сложная. Нельзя просто взять и записать в всё в один файл, как попало. Чтобы избежать неразберихи, люди договариваются о стандартах: розетки, батарейки, usb-входы. Программисты — тем более люди, они договорились о форматах и способах хранения данных.
- Цель любого формата — структурировать данные, чтобы с ними было проще эффективнее работать: чтобы сразу было понятно, где файл начинается, где хранится служебная информация, как в нём упакованы данные. Форматов и способов хранения данных много, мы познакомимся с самыми актуальными: XML, JSON и базы данных.



ФОРМАТ XML

- XML — это способ представить любую информацию в виде древовидной структуры. В жизни очень часто человек организует информацию именно в виде дерева. Давайте представим, как могла бы быть в виде XML упакована информация из вашего аккаунта в какой-нибудь социальной сети:

```
<?xml version="1.0" encoding="utf-8"?>

<account>

  <photos>
    <photo title="фоточка 1" />
    <photo title="фоточка 2" />
  </photos>

  <friends total="2">
    <friend number="1">Гоша</friend>
    <friend number="2">Петя</friend>
  </friends>

</account>

<!-- а вот так, кстати пишутся комментарии -->
```


ФОРМАТ XML

- Типичный XML файл начинается с заголовка, в котором указывается версия XML и кодировка файла `<?xml version="1.0" encoding="utf-8"?>`.
Дальше идёт дерево так называемых тегов. Корень у дерева один. Так и в XML-файле корневой тег-контейнер (оборачивающий всё остальное) — один. В нашем случае `<account></account>`.

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<account>
```

```
  <photos>
```

```
    <photo title="фоточка 1" />
```

```
    <photo title="фоточка 2" />
```

```
  </photos>
```

```
  <friends total="2">
```

```
    <friend number="1">Гоша</friend>
```

```
    <friend number="2">Петя</friend>
```

```
  </friends>
```

```
</account>
```

```
<!-- а вот так, кстати пишутся комментарии -->
```

ФОРМАТ XML

- Теги записываются в треугольных скобках и бывают открывающие (`<photos>`) и закрывающие, в начале названия которых стоит слеш (`</photos>`). Если какой-то элемент XML-структуры состоит из двух таких тегов, то он называется контейнером, т.к. в нём могут находиться другие элементы:

```
<photos>  
  <photo title="фоточка 1" />  
  <photo title="фоточка 2" />  
</photos>
```

ФОРМАТ XML

- Если же у тега нет закрывающего, то такой тег содержит слеш в самом конце: (`<photo .../>`).
- Ещё у тегов бывают атрибуты, а у атрибутов — значения. Пары атрибут-значение записываются с помощью знака равно: (`<photo title="фоточка 2" />`)
- Также тег-контейнер может содержать внутри себя кроме других тегов просто обычный текст.

```
<some_tag>  
  Я текст внутри тега "some_tag"  
</some_tag>
```


ФОРМАТ XML

- Вот вкратце и всё, что нам сейчас нужно знать про формат хранения данных XML.
- Какие именно теги будут в вашей структуре — уже определяете вы, XML лишь накладывает требования на структуру, которые мы описали выше. Скажем, нельзя открыть тег и не закрыть его, нельзя оставить открытой треугольную скобку вот так `<error_tag`.
- XML настолько популярен, что многие программы в нем хранят данные. Например, популярный редактор текста Microsoft Word: его файлы `*.docx` по сути являются zip-архивами, если такой файл переименовать, поменяв расширение на `.zip`, то можно залезть внутрь такого архива и посмотреть, какие xml-файлы там лежат.

ПРОЕКТИРУЕМ ПРОГРАММУ ДЛЯ УЧЕТА РАСХОДОВ

- Давайте напишем программу, которая будет показывать, сколько на что мы потратили, а также выводить сумму всех трат за каждый месяц. Для этого нам как-то надо хранить данные о покупках.
- Для начала давайте придумаем структуру, в которой мы будем описывать информацию о тратах. Этот XML-файл пока будем редактировать руками, а на позже научимся делать это с помощью программы.

ПРОЕКТИРУЕМ ПРОГРАММУ ДЛЯ УЧЕТА РАСХОДОВ

- expenses.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<expenses>
  <expense date="9.7.2021" category="Образование" amount="900" >
    Книжка по Ruby on Rails
  </expense>

  <expense date="1.7.2021" category="Запись стрима" amount="400" >
    Петличка и провода
  </expense>

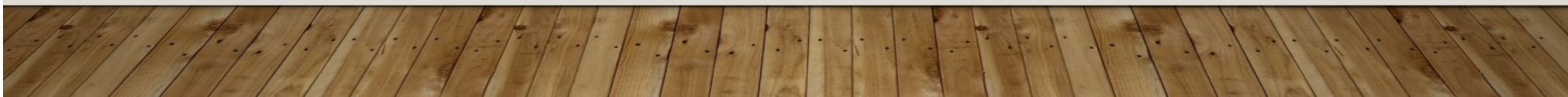
  <expense date="23.6.2020" category="Запись стрима" amount="3500" >
    Софтбокс + штатив
  </expense>
</expenses>
```

ПРОЕКТИРУЕМ ПРОГРАММУ ДЛЯ УЧЕТА РАСХОДОВ

- Вот так мы будем хранить эти данные. Как вы видите, у нас есть корневой тег **expenses**, в него вложены теги **expense** (по одному на каждую трату), внутри каждого такого тега, текст с информацией о покупке, а в атрибутах **date**, **amount** и **category** соответственно дата, сумма и категория покупки.

ПРОЕКТИРУЕМ ПРОГРАММУ ДЛЯ УЧЕТА РАСХОДОВ

- Для того, чтобы удобно работать с XML-файлами во всех современных языках есть парсеры.
- **Парсер** — это инструмент (чаще всего библиотека) для чтения данных из текста и представления их в удобной для работы с ними форме.
- Обычно их в каждом языке даже несколько. Какие-то быстрее, какие-то удобнее. Один из таких парсеров (**REXML**) уже встроен в Ruby (начиная с версии 1.9). Им и воспользуемся.



ПРОГРАММА ДЛЯ УЧЕТА РАСХОДОВ

- Сейчас мы напишем программу, которая будет идти по всем тратам и добавлять их в ассоциативный массив, в котором в виде ключа будут даты: то есть все траты, сделанные в один и тот же день, попадут в один и тот же элемент.

1 января → expense...

5 января → expense...

⋮

30 января → ...

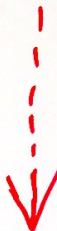
2 февраля → expense...

⋮


ПРОГРАММА ДЛЯ УЧЕТА РАСХОДОВ

- Затем мы пройдемся по всем дням и объединим их в месяцы:

1 января → expense...
5 января → expense...
⋮
30 января → ...
2 февраля → expense...
⋮

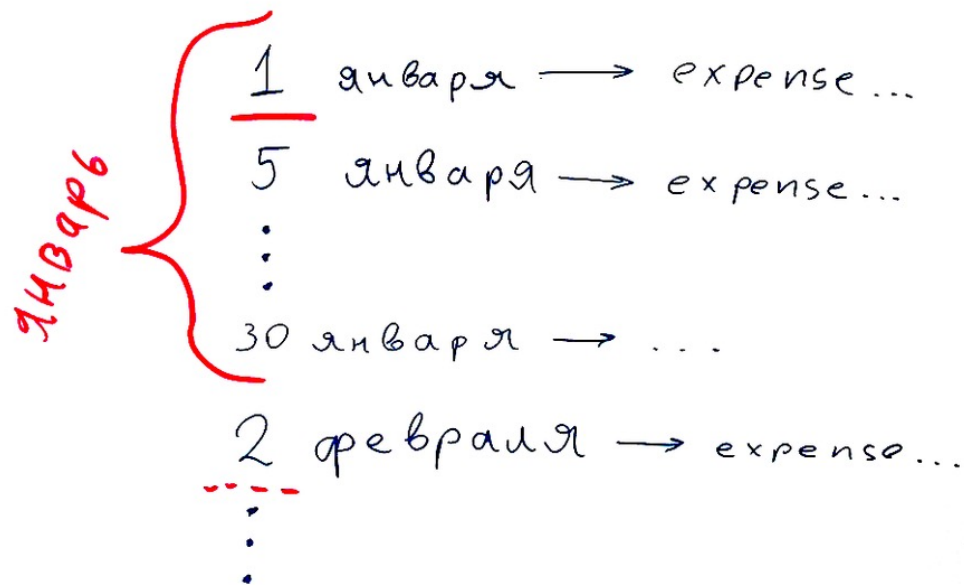


1 января → expense...
5 января → expense...
⋮
30 января → ...
2 февраля → expense...
⋮



ПРОГРАММА ДЛЯ УЧЕТА РАСХОДОВ

- И, наконец, после того, как у нас будут все месяцы с полным набором трат, мы можем вывести пользователю, сколько денег он потратил в каждом месяце и на что.



Handwritten list of expenses with red annotations:

- январь** (written vertically in red) is grouped by a red bracket next to the first three items.
- 1 января → expense...
- 5 января → expense...
- ⋮
- 30 января → ...
- 2 февраля → expense...
- ⋮

A red dashed arrow points downwards on the right side of the list.

ПРОГРАММА ДЛЯ УЧЕТА РАСХОДОВ

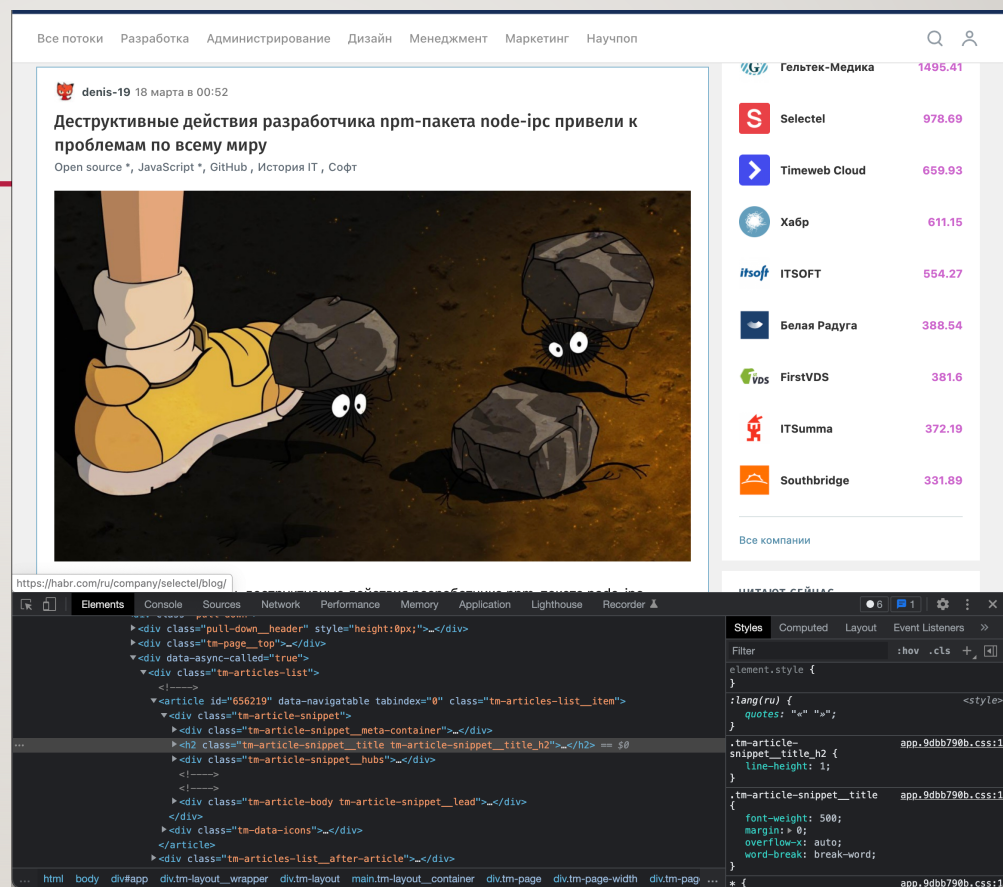
- Давайте писать программу!
- Листинг `expenses_reader.rb` приведен в папке с уроком.

НЕМНОГО О HTML

- HTML — грубо говоря, одна из разновидностей XML. Требования к разметке в некоторых версиях HTML, однако (а версий HTML много, сейчас, например, самая актуальная HTML5), могут существенно отличаться от того, что мы рассказали про XML.

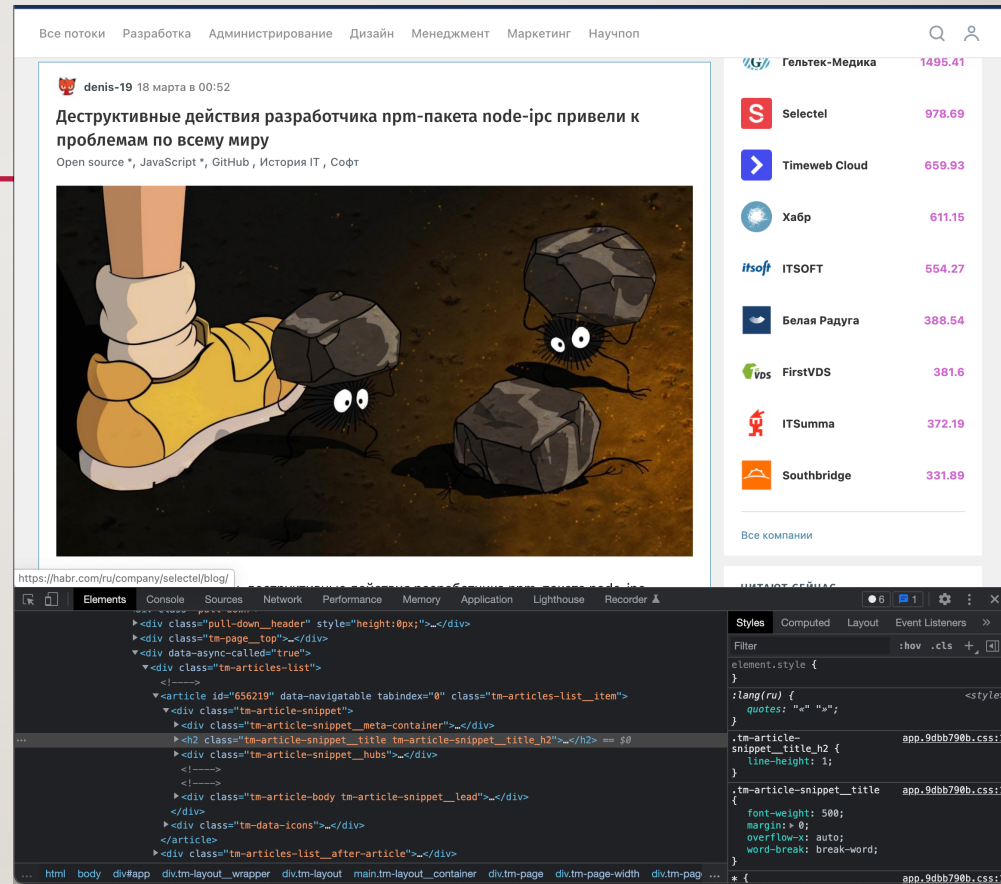
НЕМНОГО О HTML

- Давайте зайдём на сайт habr.ru и посмотрим его html-разметку. Для этого нажмите на любой неактивный элемент на сайте правой кнопкой мыши и выберите пункт «Исходный код страницы».



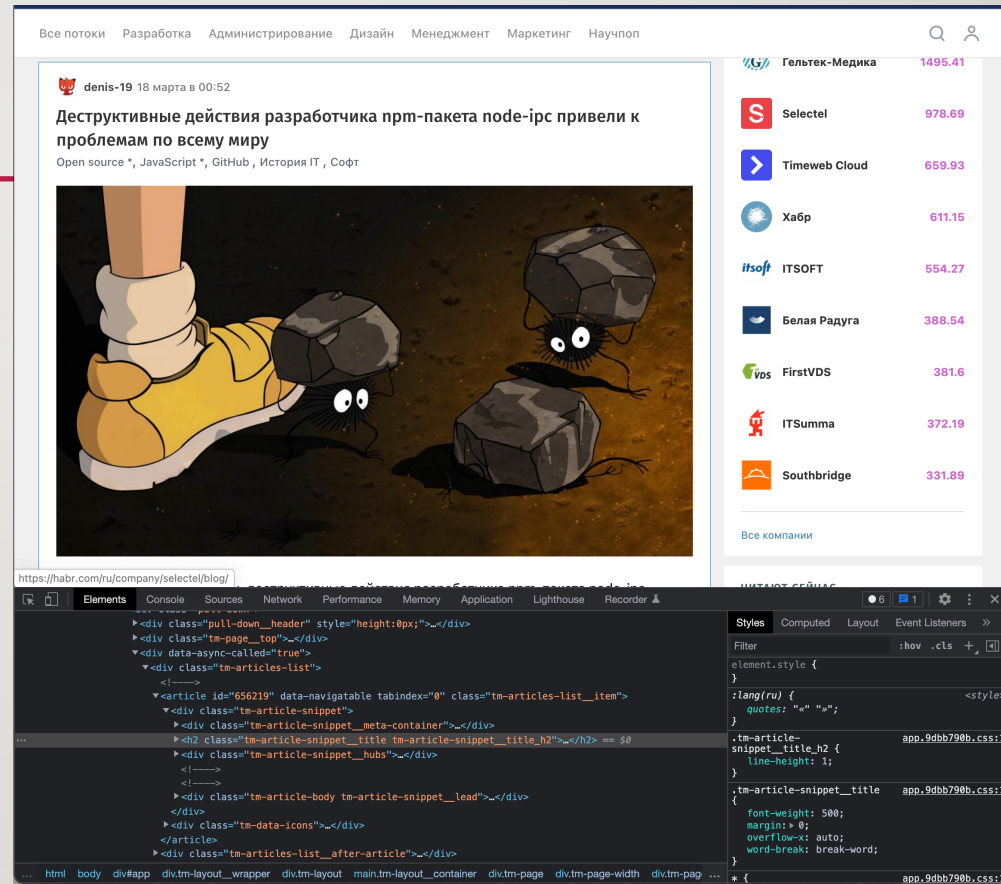
НЕМНОГО О HTML

- Те блоки гипертекста, которые вы видите, вам должны показаться знакомыми — тоже теги, вложенные друг в друга, атрибуты и так далее. Нам это знание пригодится в последующих уроках, когда будем работать с сетью.



НЕМНОГО О HTML

- Итак, мы узнали о формате XML, научились читать данные из XML-файлов и узнали немного про HTML. На следующем уроке мы научимся писать свои XML файлы и сделаем так, чтобы нам не приходилось добавлять траты в файл **expenses.xml** руками.



ЧТЕНИЕ XML-ВИЗИТКИ

- Сделайте свою визитку в формате XML.
- Укажите свои фамилию, имя, отчество, телефон, адрес электронной почты и немного напишите о своих навыках.
- Напишите программу, которая читает эту визитку в формате XML и красиво выводит информацию на экран.
- **Например:**

Иван А. Попов
+7 999 100-30-20, ivan.popov@mail.ru
Начинающий программист на Ruby

ЧТЕНИЕ XML-ВИЗИТКИ. ПОДСКАЗКА

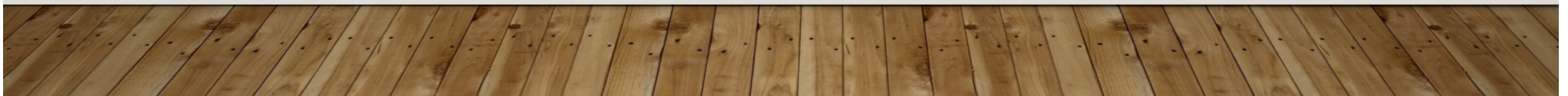
- Создайте файл как в уроке. Начните со служебной конструкции:

```
<?xml version="1.0" encoding="utf-8"?>
```

- Помните, что корневой элемент только один. В программе откройте файл с помощью библиотеки **rexml** и доставайте все нужные поля с помощью метода **elements**.

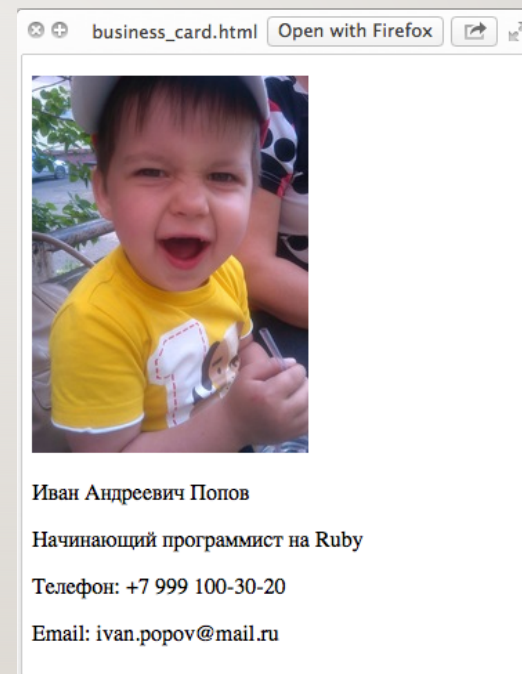
```
doc.root.elements["first_name"].text
```

- Возможно, вам понадобится заглянуть в [инструкцию по использованию этой библиотеки](#), описание [класса REXML](#).



XML-ВИЗИТКА

- На уроке мы показывали, как устроен любой сайт.
- Используя самые простые теги: `<body>`, `<p>` и `` сделайте простенькую страничку с вашей визиткой в формате html. Но обязательно с фотографией.



XML-ВИЗИТКА. ПОДСКАЗКА

- Помните, что html — это просто xml, построенный по определённым правилам.
- Начните с корневого тега `<html>`, добавьте в него обязательный тег `<body>`, а уже в нём напишите в параграфах `<p>` информацию о себе и вставьте фото с помощью тега ``.
- Как пользоваться всеми этими тегами легко найти в Интернете. В качестве заготовки возьмите вот этот код:

```
<!DOCTYPE HTML>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
  </head>
  <body>
    Тут будет текст вашей визитки
  </body>
</html>
```

XML-МАГАЗИН

- В домашнем задании к прошлому уроку мы писали магазин, который торговал музыкой, фильмами и книжками. Не удобно и не правильно создавать их руками прямо в коде программы.
- Научите магазин читать данные из отдельного XML-файла.

XML-МАГАЗИН

- **Например:**

```
>ruby main.rb
```

Что хотите купить?

0: Книга «Приключения Тома Сойера», автор: Марк Твен – 1000 руб. [осталось: 5]

1: Книга «Воспоминания Шерлока Холмса», автор: Артур Конан Дойль – 1100 руб. [осталось: 1]

2: Диск Judas Priest – Turbo (Heavy Metal) – 500 руб. [осталось: 2]

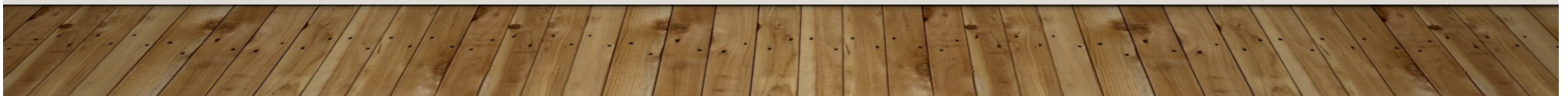
3: Диск Judas Priest – Ram It Down (Heavy Metal) – 550 руб. [осталось: 3]

4: Фильм «Пролетая над грездом кукушки», реж. Милош Форман (1975) – 350 руб. [осталось: 3]

x. Покинуть магазин

x

Спасибо за покупки, с Вас 0 руб.



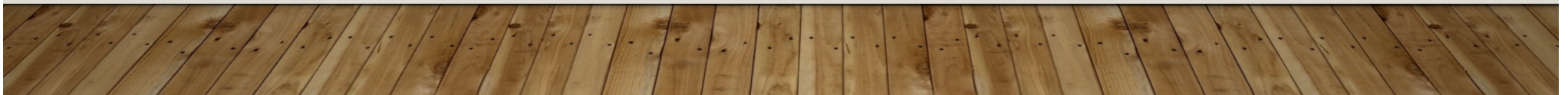
XML-МАГАЗИН. ПОДСКАЗКА

- Для начала придумайте, в каком формате вы будете хранить ваши товары.
- Предлагаем в таком:

```
<?xml version="1.0" encoding="utf-8"?>
<products>
  <product price="1000" amount_available="5">
    <book title="Приключения Тома Сойера" author_name="Марк Твен"></book>
  </product>
</products>
```


XML-МАГАЗИН. ПОДСКАЗКА

- Потом напишите у класса **Product** статический метод **read_from_xml**, который в качестве параметра принимает, например, имя файла со списком продуктов и передает его парсеру **REXML** как мы это делали на уроке.
- Любым удобным способом ([документация REXML](#)) достаньте из соответствующих тегов информацию о товаре и создайте экземпляры соответствующего класса на каждую запись в xml-файле.
- А затем верните массив товаров.



СПРАВОЧНАЯ ИНФОРМАЦИЯ

- [XML для чайников](#)
- [Как работать с парсером REXML \(eng\)](#)
- [Основы XML для начинающих](#)
- [XML для начинающих](#)
- [Основы HTML](#)
- [HTML5Book](#)

СПАСИБО ЗА ВНИМАНИЕ!

Лекция 27. Хранение данных, XML, HTML