TAU TECHNOLOGIES

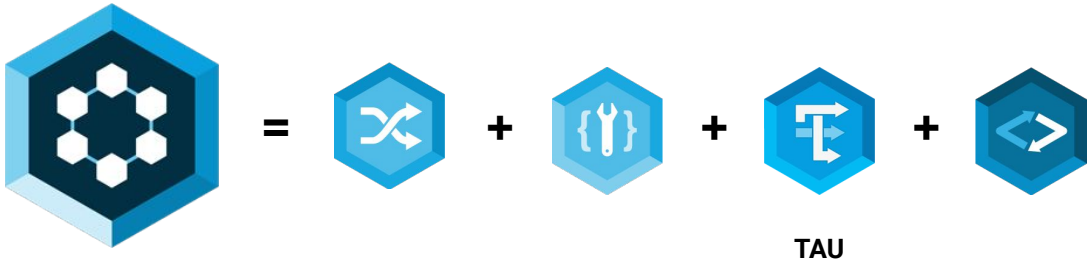
RhoMobile

кросс-платформенная разработка для мобильных и настольных устройств на iOS, Android, WinCE/WM, Sailfish/Аврора, Alt Linux, Astra Linux, Rosa Linux, Red OS, Linux, Windows, macOS

О чем пойдет речь

- О продуктах компании Tay Технологии
- Ключевые факты о решении RhoMobile и компании Tay Технологии
- О кросс-платформенной разработке и ее месте в корпоративном секторе
- Архитектура кросс-платформенного решения RhoMobile (для разработчиков)
- Установка RhoMobile на все системы (для разработчиков)
- Подробное описание разработки Rhodes приложения на примере (для разработчиков)

О продуктах компании Тау Технологии



- Полностью свободное и открытое решение для разработки кросс-платформенных нативных мобильных приложений с использованием веб-технологий HTML, CSS, JS и Ruby.
- Поддерживаются iOS, Android, Windows CE/Mobile, SailFish/Аврора, Alt Linux, Astra Linux, ROSA Linux, Red OS, Windows Desktop, Linux, macOS.
- RhoConnect - решение для синхронизации данных между мобильным приложением и корпоративным бэкендом.
- RhoStudio - интегрированная среда разработки
- Tau Extensions - дополнительные модули, в том числе реализация Node.js на мобильном устройстве, собственный порт WebKit для WinCE/WM, поддержка Crosswalk, OpenSSL и многое другое.

RhoBrowser

RHO BROWSER

RhoBrowser это мощный современный промышленный браузер, который позволяет разработчикам создавать веб приложения с богатым функционалом, недоступным в обычном браузере. И все это не только на современных платформах (iOS, Android, Sailfish/Аврора и настольные платформы), но также и на устаревших WinCE/WM устройствах !

Мобильные веб приложения для RhoBrowser получают доступ к функциональности устройств и периферии — сканирование различных видов кодов, работа с Bluetooth принтерами, ввод рукописной подписи и многое другое.

Но что еще более важно — мы готовы кастомизировать и интегрировать RhoBrowser в вашу информационную систему.

Кросс-платформенность

Поддержка iOS, Android, WM, WinCE, Sailfish/Аврора
Поддержка настольных платформ

Barcode/RFID сканеры

Распознавание всех видов кодов с помощью камеры на любом устройстве
Работа с аппаратными сканерами кодов

HTML/CSS/JS

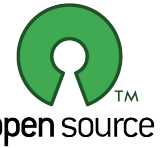
Современные HTML5-CSS3-JS решения в том числе и на WM/CE устройствах благодаря собственному портированному WebKit

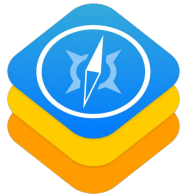
Настройка

Кастомизация под ваши требования:
установка веб приложений на сервер или устройство.
Управление установкой приложений на устройстве.
Другие возможности

Преимущества RhoMobile Suite

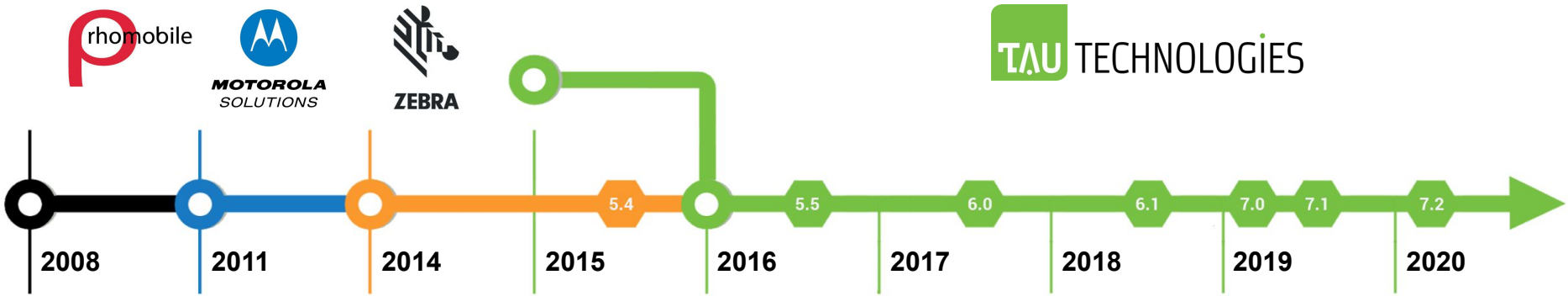
- Ruby on Rails среда прямо на мобильном устройстве!
- Обширный набор API, позволяющий полностью использовать возможности устройства, доступный как из Ruby, так и из Javascript.
- Возможность переносить HTML/CSS/Javascript код между веб приложениями, Cordova/PhoneGap и RhoMobile.
- Поддерживает функционал всех индустриальных устройств Zebra Technologies.
- Полная кросс-платформенность, включая поддержку и настольных платформ.
- Возможность использования HTML5 на WM/CE с помощью собственной реализации WebKit.
- На рынке более 10 лет. Оперативное обновление продукта для поддержки новых версий платформ.





Ключевые факты из истории RhoMobile и Tay Технологии

2008 Основан стартап RhoMobile
2011 RhoMobile приобретен компанией Motorola Solutions.
2014 RhoMobile приобретен компанией Zebra Technologies как часть Motorola enterprise бизнеса.
2015 Несколько участников RhoMobile основали компанию Tau Technologies.
2016 Zebra полностью перевела RhoMobile в open-source (ранее были закрытые части)
RMS 5.4 последняя версия выпущенная Zebra.
Tau становится технологическим партнером Zebra и получает обратно права на основной репозиторий продукта на GitHub (https://github.com/rhomobile/rhodes).
Подробнее: Zebra: RhoMobile Open Source FAQ
RMS 5.5 выпущена Tau в сентябре 2016.
2017-2019 Tau выпустила RMS версий 6.0, 6.1, 7.0 и 7.1
2020 RMS 7.2, 7.3 выпущена Tau



О кросс-платформенной разработке

Кросс-платформенная разработка

- снижение стоимости разработки
- быстрая разработка сразу на все платформы
- Особенности ОС поддерживаются разработчиками кросс-платформенного решения
- небольшие размеры команды
- переиспользование веб разработчиков и кода (для гибридных решений включая Rhodes)
- переиспользование Ruby on Rails разработчиков и кода (Rhodes)

- относительно низкая производительность (не актуально для корпоративного сектора)
- несовременный UI (решается использованием современных Javascript фреймворков в гибридных решениях)
- зависимость от разработчика решения (менее актуально в случае полностью open-source решения)

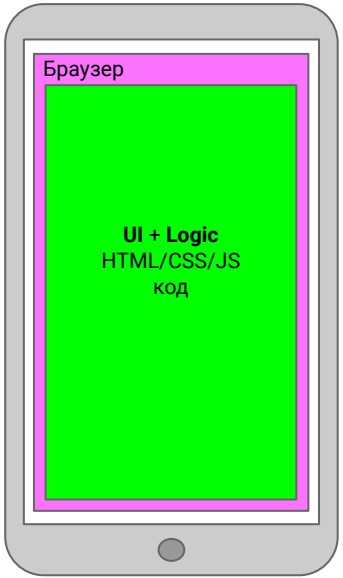
Отличный выбор для корпоративной разработки !

Архитектуры мобильных кросс-платформенных приложений

Простое не кросс-платформенное нативное приложение

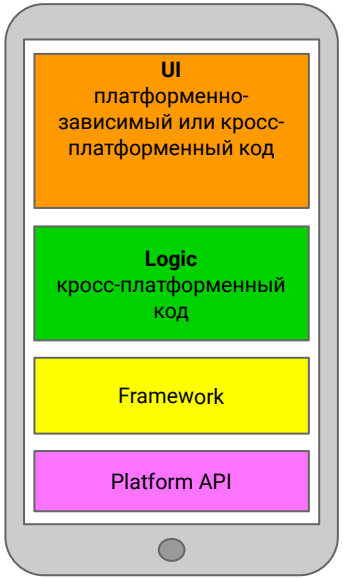
iOS: ObjC, Swift, C++
Android: Java, C++
WinCE/WM: C#, C++

Web кросс-платформенное приложение



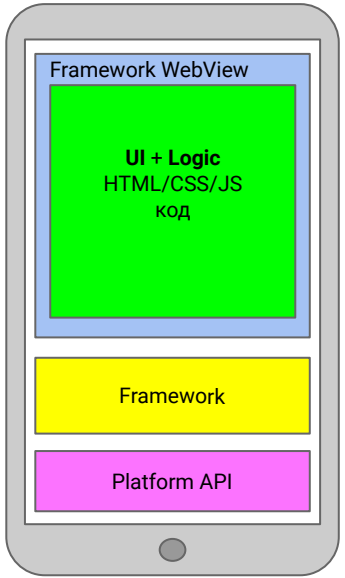
a lot of
HTML/CSS/JS фреймворков

Нативное кросс-платформенное приложение



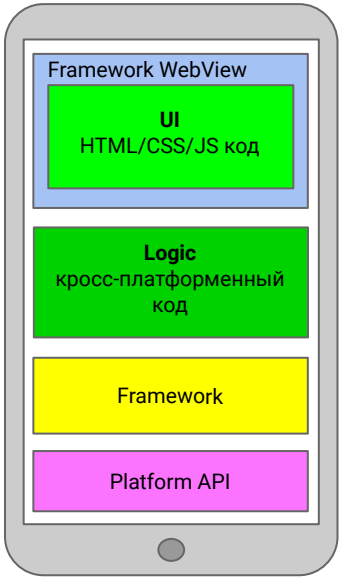
Xamarin (C#)
Appcelerator (JS)
React Native (JS)
NativeScript (JS)
QT (C++, QML)
RubyMotion (Ruby)
CodenameOne (Java)
Corona (Lua)
Flutter (Dart)

Гибридное кросс-платформенное приложение



Cordova/PhoneGap RhoMobile
(в чисто гибридном режиме)
+ а множество
HTML/CSS/JS фреймворков

Смешано-гибридное кросс-платформенное приложение

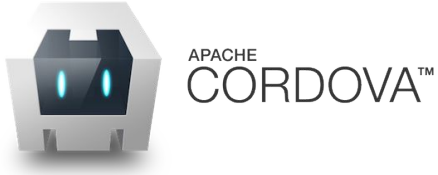


RhoMobile (Ruby)
+ множество
HTML/CSS/JS фреймворков

Решения для мобильной кросс-платформенной разработки

Нативные

Гибридные



RHOMOBILE SUITE

Некоторые наши доклады о кросс-платформенной мобильной разработке

- SECR 2018, Разработка нативных и гибридных приложений для Sailfish Mobile OS RUS (совместное выступление с компанией ОМП)

  https://2018.secrus.org/lang/ru/program/submitted-presentations/development-hybrid-cross-platform-applications-for-sailfish-os/

- SECR 2017, Развитие гибридных решений для разработки кросс-платформенных мобильных приложений

  http://2017.secrus.org/lang/ru/program/submitted-presentations/improvement-of-hybrid-solutions-for-the-development-of-cross-platform-mobile-applications

- Node.js SPb Meetup 2017, Разработка кросс-платформенных мобильных гибридных приложений на базе Node.js с использованием RhoMobile

  Видео: https://youtu.be/Yxsf06JSGCE?t=6634

  Презентация: http://files.tau-platform.com/Events/2017_03_31_SPB_Nodejs_meetup/RhoMobile_SPB_Nodejs_meetup_2017_03_31_RUS.pdf

- SECR 2016, Настоящее и будущее решений для разработки кросс-платформенных мобильных гибридных приложений в корпоративной сфере

  http://2016.secrus.org/lang/ru/program/submitted-presentations/current-state-and-future-of-solutions-for-develop-enterprise-cross-platform-mobile-applications

- Rails Club 2016, RhoMobile. Разработка кросс-платформенных мобильных гибридных приложений.

  Видео: https://youtu.be/wPJXNo4kdfg

  Презентация: http://files.tau-platform.com/Events/2016_10_RailsClub/Rhomobile_RailsClub_2016_full_RUS.pdf

Архитектура решения Rhodes

Язык Ruby и Ruby on Rails

Ruby

Ruby - динамический, рефлективный, интерпретируемый высокоуровневый язык программирования.
Язык обладает независимой от операционной системы реализацией многопоточности, сильной динамической типизацией, сборщиком мусора и многими другими возможностями.
Разработан Юкихиро Мацумото в 1995 году.
Входит в дистрибутивы Linux, MacOS и др.
Активно развивается и поддерживается.

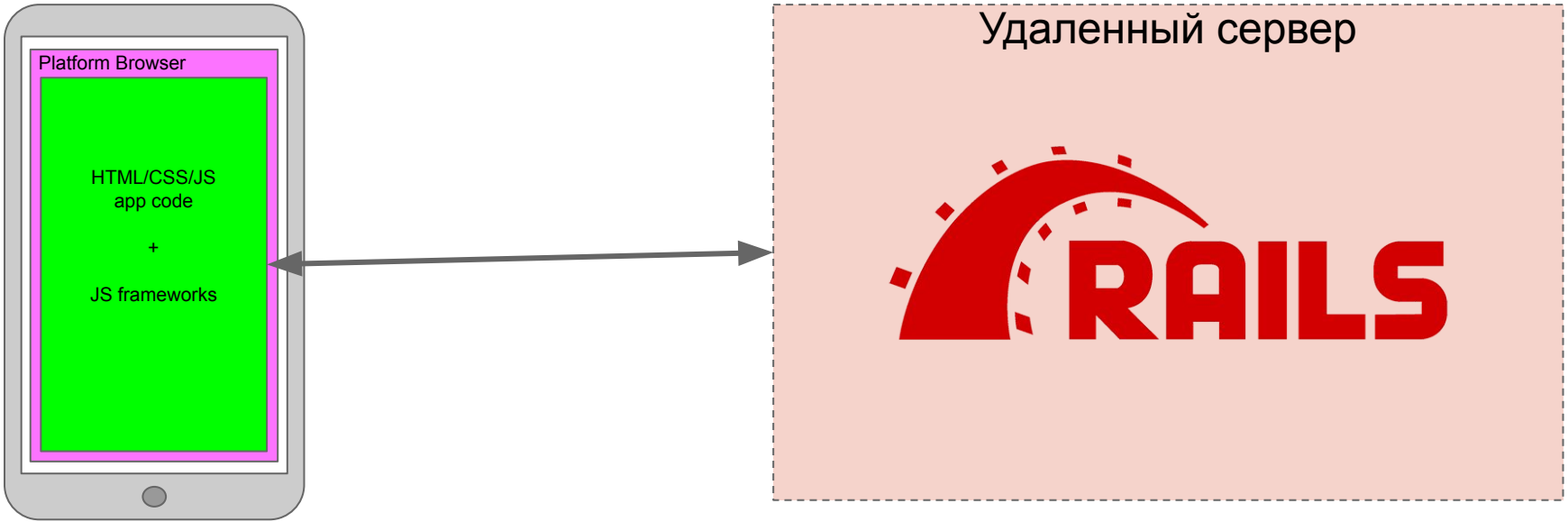
Сайт - https://www.ruby-lang.org/ru/

Ruby on Rails (RoR) - фреймворк, написанный на языке программирования Ruby, реализует архитектурный шаблон Model-View-Controller для веб-приложений, а также обеспечивает их интеграцию с веб-сервером и сервером баз данных. Является открытым программным обеспечением и распространяется под лицензией MIT.
Создан Давидом Хейнемейером Ханссоном в 2005 году.
Активно развивается и поддерживается.

Сайт - https://rubyonrails.org/

Client-Server web приложение с удаленным сервером на Ruby on Rails

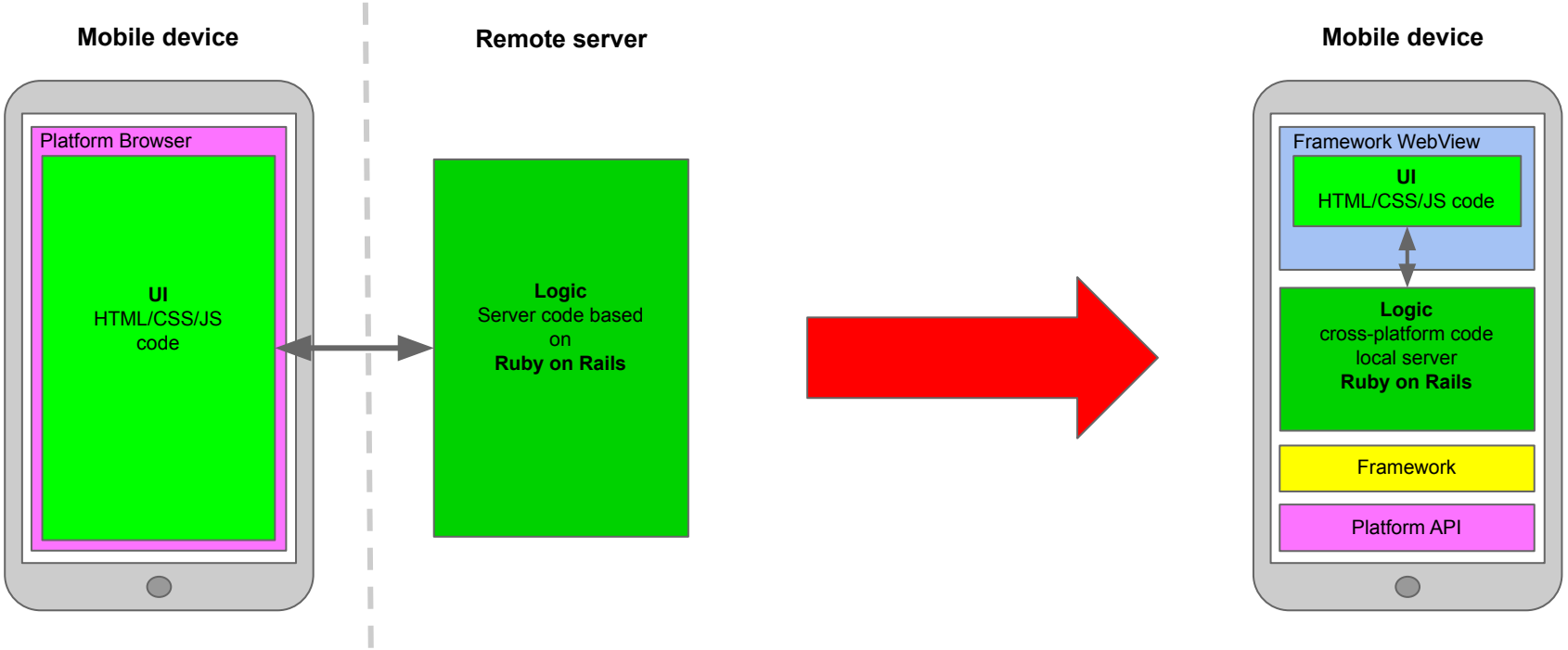
Рассмотрим типовое веб приложение с удаленным сервером.
У нас есть браузер с HTML/CSS/JS кодом и удаленный сервер с кодом для работы с данными и логикой. В данном случае сервер реализован на популярном решении Ruby on Rails. Это весьма распространенное решение. А значит :
У нас много готового кода, документации и т.п. для этого решения
У нас много опытных разработчиков для этого решения
Мы можем использовать это решение также для портала и т.п.



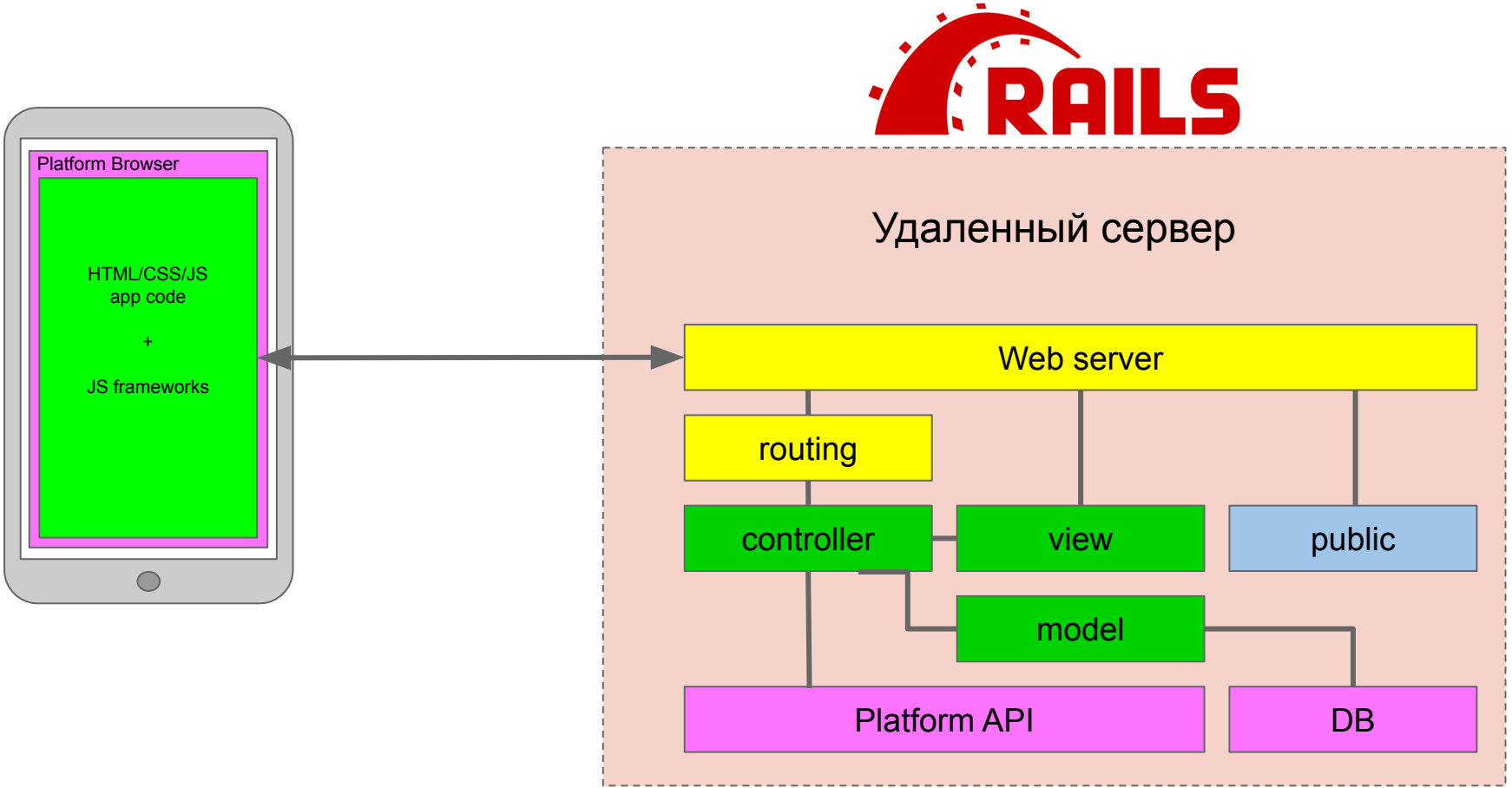
А что если перенести удаленный сервер прямо на устройство ?

Веб приложение с удаленным сервером

Нативное приложение со смешанно-гибридной архитектурой



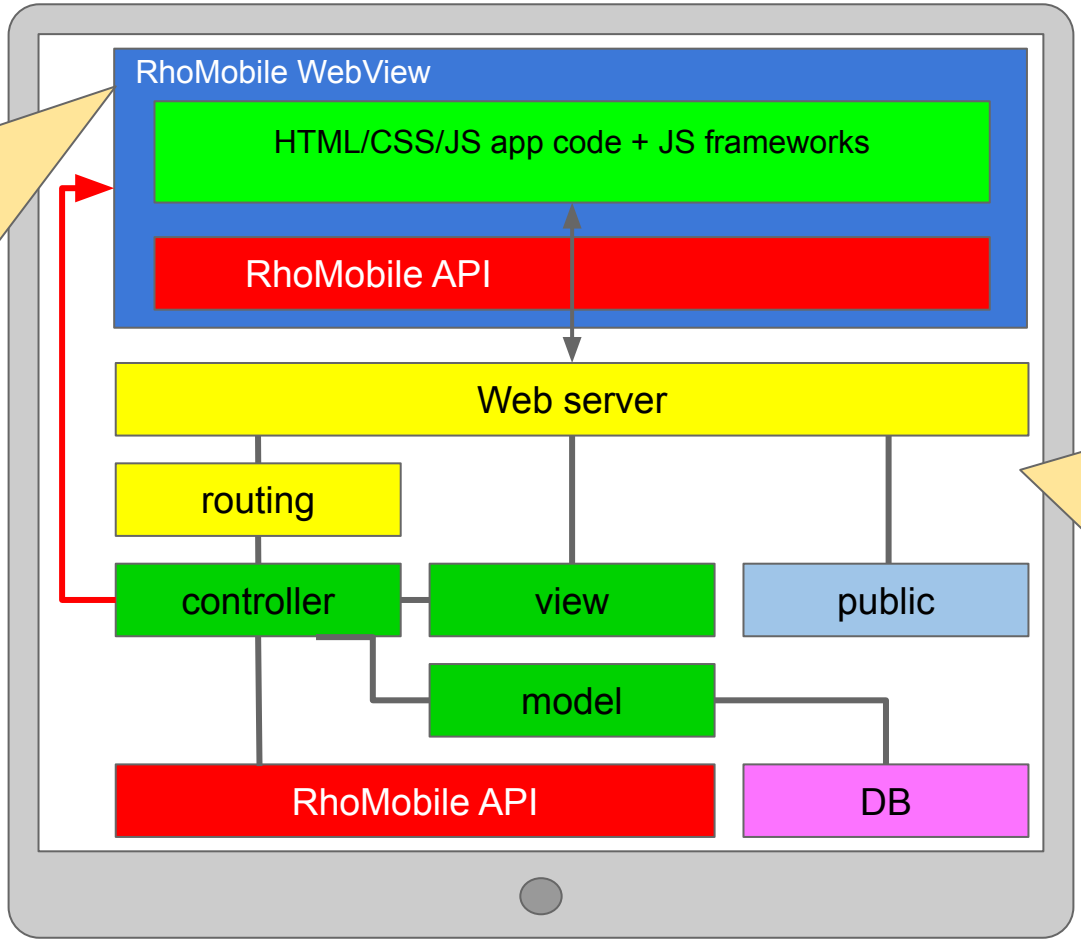
Веб приложение с удаленным сервером на базе Ruby on Rails



Rhodes приложение с WebView и локальным RoR-подобным сервером

Гибридная архитектура. UI выполнен в WebView на базе веб технологий (HTML/CSS/JS).

Разработчики могут свободно применять любые имеющиеся на рынке Javascript фреймворки.



RHODES

Мы имеем локальный HTTP сервер с полноценной Ruby Виртуальной Машиной. Ruby код непосредственно выполняется прямо на мобильном устройстве

Почему так “сложно” ?

Вам кажется что слишком сложно переносить серверные технологии на устройство и отделять UI от логики в разных средах - UI в среде WebView (по сути аналогичном обычному браузеру), а логика в среде Ruby с похожим на Ruby on Rails фреймворком ?
Но если подумать, то у такого решения есть много преимуществ :

- Обе среды хорошо знакомы, особенно WebView(HTML/JS/CSS) - веб разработчиков очень много на рынке, в любом корпоративном решении наверняка есть и веб решения, с веб приложениями и тп, и все эти наработки и даже напрямую код, можно использовать при создании и нативного приложения. Само собой что и веб разработчики смогут, без переподготовки, принять участие в разработке приложений.
- Что касается Ruby и Ruby on Rails, то эта среда конечно менее распространена по сравнению с HTML/JS, но тоже весьма популярна среди серверных разработчиков. Язык Ruby довольно простой, но мощный ООП язык, очень удобен при работе с моделями баз данных, скриптования бизнес логики и тп.
- Таким образом соединяя хорошо известные среды с огромным количеством готовых решений, кода, примеров, большим комьюнити мы получаем решение для разработки приложений с единым кодом сразу для всех основных операционных систем, при этом разработчикам не нужно вникать в особенности конкретных систем + можно использовать веб наработки, веб разработчиков и серверных разработчиков без переучивания.

Установка RhoMobile

Установка RhoMobile (общее)

Перед установкой пожалуйста прочтите о необходимых требованиях к системе. Подробности:
http://docs.tau-platform.com/en/7.3/guide/rhomobile-install
http://docs.tau-platform.com/en/7.3/guide/nativesdksetup

Поскольку RhoMobile изначально базировался на Ruby, то для работы необходимо наличие установленного Ruby в системе. Ruby имеет собственный менеджер пакетов(библиотек) RubyGems, а пакеты называются gems. Это очень похоже на модули в Node.js.
Само решение RhoMobile поставляется в виде набора пакетов для Ruby - gem-ов.
Основные пакеты - **rhodes** и **rhoconnect-client**. Также есть серверное решение - **rhoconnect** и набор расширений **rho-tau-extensions**.
Для Windows и macOS имеется единый инсталлятор, но для остальных систем нужно ставить gem-ы вручную. После установки gem-ов в командной строке станут доступны Rhodes команды.
Также практически все разработчики Ruby используют менеджер версий Ruby - RVM, поэтому мы настоятельно советуем использовать его.

Мы рекомендуем скачать последнюю версию RMS 7.3 с нашего сайта :
https://tau-platform.com/en/developers/downloads/

После установки настройте пожалуйста пути к необходимым платформенным SDKs(необходимо для Android, для других платформ можно пропустить):

```
$ rhodes-setup
```

Ruby Gems и RVM

RubyGems - система управления пакетами для языка программирования Руби, которая предоставляет стандартный формат для программ и библиотек Руби (в самодостаточном формате «gems»), инструменты, предназначенные для простого управления установкой «gems», и сервер для их распространения.
Входит в обычную поставку Ruby.
Традиционно используется для продуктов Ruby.
RhoMobile поставляется в виде пакетов (gems).
Сайт - https://rubygems.org/

RVM (Ruby Version Manager) - утилита командной строки для установки, управления и переключения разными версиями Ruby и наборов пакетов (gemsets). Используется подавляющим большинством разработчиков Ruby.
Настоятельно рекомендуется к установке и использованию при работе с RhoMobile.
Сайт - https://rvm.io/

Установка RhoMobile на macOS

Рекомендуем установить прежде RVM и поставить какую либо из свежих версий Руби - рекомендуется 2.5.

Скачайте DMG архив нужной версии с http://rhomobile.tau-platform.com/index.html
Текущая актуальная стабильная версия 7.3 (или берите более свежую 7.3.*)

Откройте DMG и установите все пакеты gem в нужный gemset RVM при помощи скрипта “Install Gems” - рекомендуем проводить установку gems без root(под sudo) прав.

Также необходимо установить XCode для разработки под iOS и/или Android SDK и NDK для разработки под Android.

Настройте необходимые пути, в том числе, к Android SDK/NDK при помощи команды

```
$ rhodes-setup
```

Установка RhoMobile на Windows

Скачайте инсталлятор *.exe и запустите его.
Текущая актуальная стабильная версия 7.3 - файл RMS_7.3.exe
(или берите более свежую 7.3.*)
Адрес для скачивания - http://rhomobile.tau-platform.com/index.html

Инсталлятор содержит в себе и установит все необходимое включая Ruby, необходимые ruby gems и т.п.

Настройка системы для разработки под мобильную Аврора ОС

Установите RhoMobile.
Установите среду разработки “Аврора”:
https://community.omprussia.ru/documentation/software_development/sdk/installation.html
Увеличьте объем оперативной памяти на виртуальной билд-машине минимум до 2-х гигабайт

Из-за особенностей реализации сборки в виртуальной машине, папка с кодом приложения и папки с rhodes и rhoconnect-client должны находится в домашней директории. Вы можете скопировать их из папки где находятся ruby gems (команда “get-rhodes-info --rhodes-path” вернет путь к установленному rhodes gem)
или непосредственно из репозиториев rhodes (https://github.com/rhomobile/rhodes) и rhoconnect-client (https://github.com/rhomobile/rhoconnect-client)

Настраиваем пути к виртуалке и SDK Авроры в конфигурационном файле rhodes - rhobuild.yml (находится в корне папки rhodes). Для Windows это например :

```
env:
  paths:
    sailfish: C:/AuroraOS
    vbox: C:/Program Files/Oracle/VirtualBox
```

Установка RhoMobile на Ubuntu (проверено на 20.04)

Скачайте два основных gems из RMS 7.3 или более свежей стабильной версии (7.3.x)
http://rhomobile.tau-platform.com/index.html

Выполните следующие команды:

```
sudo apt-get install gcc g++ make wget ruby ruby-dev

sudo apt-get install build-essential

sudo apt-get install \
"^libxcb.*" libx11-dev libx11-xcb-dev libxcursor-dev libxrender-dev libxrandr-dev \
libxext-dev libxi-dev libxss-dev libxt-dev libxv-dev libxxf86vm-dev libxinerama-dev \
libxkbcommon-dev libfontconfig1-dev libharfbuzz-dev \
libasound2-dev libpulse-dev libdbus-1-dev udev mtdev-tools webp \
libudev-dev libglm-dev libwayland-dev libegl1-mesa-dev mesa-common-dev \
libgl1-mesa-dev libglu1-mesa-dev libgles2-mesa libgles2-mesa-dev libmirclient-dev \
libproxy-dev libgtk2.0-dev libgtk-3-dev libcups2-dev

sudo apt-get install qtdeclarative5-dev qtmultimedia5-dev libqt5multimediawidgets5 \
libqt5multimedia5-plugins libqt5multimedia5 qtwebengine5-dev libx11-xcb-dev \
libglu1-mesa-dev libxrender-dev libfontconfig1-dev

grep -qxF 'export QTDIR=/usr/lib/x86_64-linux-gnu/qt5' ~/.bashrc || echo 'export QTDIR=/usr/lib/x86_64-linux-gnu/qt5' >> ~/.bashrc

source ~/.bashrc
```

Установите RhoMobile gems:

```
sudo gem install rhodes-7.2.gem
sudo gem install rhoconnect-client-7.2.gem
```

Установка RhoMobile на Alt Linux (проверено на 9.0)

Скачайте два основных gems из RMS 7.3 или более свежей стабильной версии (7.3.x)
http://rhomobile.tau-platform.com/index.html

Выполните следующие команды:

```
sudo apt-get update

sudo apt-get install git gcc gcc-c++ ruby libruby-devel irb gmp-devel libGL-devel libGLU-devel rpm-build zlib-devel

sudo apt-get install qt5-base-devel qt5-webengine-devel qt5-multimedia-devel libglfw libglfw-devel

grep -qxF 'export QTDIR=/usr/share/qt5' ~/.bashrc || echo 'export QTDIR=/usr/share/qt5' >> ~/.bashrc

source ~/.bashrc
```

Установите RhoMobile gems:

```
sudo gem install rhodes-7.2.gem
sudo gem install rhoconnect-client-7.2.gem
```

Переустановите rake:

```
sudo gem install rake
```

Установка RhoMobile на Astra Linux (проверено на Orel 2.12.29)

Скачайте два основных gems из RMS 7.3 или более свежей стабильной версии (7.3.x)
http://rhomobile.tau-platform.com/index.html

Выполните следующие команды:

```
sudo apt update

sudo apt upgrade

sudo apt install git gcc g++ make ruby-dev libxslt-dev libxml2-dev zlib1g-dev libz-dev libiconv-hook1 libiconv-hook-dev libqt5webenginewidgets5 libqt5webengine5 libqt5webenginecore5
libqt5multimedia5 libqt5multimediawidgets5 libqt5gui5 libqt5concurrent5 libqt5network5 qt5-default qt5-qmake qttools5-dev qtwebengine5-dev qtmultimedia5-dev

grep -qxF 'export QTDIR=/usr/lib/x86_64-linux-gnu/qt5' ~/.bashrc || echo 'export QTDIR=/usr/lib/x86_64-linux-gnu/qt5' >> ~/.bashrc

source ~/.bashrc
```

Установите RhoMobile gems:

```
sudo gem install rhodes-7.2.gem
sudo gem install rhoconnect-client-7.2.gem
```

Установка RhoMobile на Red OS (проверено на MUROM 7.2)

Скачайте два основных gems из RMS 7.3 или более свежей стабильной версии (7.3.x)
http://rhomobile.tau-platform.com/index.html

Выполните следующие команды:

```
sudo yum install git gcc gcc-c++ ruby ruby-devel ruby-irb gmp-devel mesa-libGL-devel mesa-libGLU-devel rpm-build

sudo yum install qt5-qtbase qt5-devel qt5-qtwebengine-devel qt5-qtmultimedia-devel glfw glfw-devel

sudo gem install io-console
sudo gem install rdoc --no-document
sudo gem install bigdecimal

grep -qxF 'export QTDIR=/usr/lib64/qt5' ~/.bashrc || echo 'export QTDIR=/usr/lib64/qt5' >> ~/.bashrc

source ~/.bashrc
```

Установите RhoMobile gems:

```
sudo gem install rhodes-7.2.gem
sudo gem install rhoconnect-client-7.2.gem
```

Установка RhoMobile на Rosa Linux (проверено на R11 EE 2016.1 Desktop)

Скачайте два основных gems из RMS 7.3 или более свежей стабильной версии (7.3.x)
http://rhomobile.tau-platform.com/index.html

Выполните следующие команды:

```
sudo urpmi git gcc gcc-c++ ruby ruby-devel libgmp-devel lib64gl-devel lib64glu-devel rpm-build lib64gmp-devel

sudo urpmi qt5-devel lib64qt5webenginecore-devel lib64qt5webenginewidgets-devel lib64qt5multimedia-devel libglfw-devel

grep -qxF 'export QTDIR=/usr/lib64/qt5' ~/.bashrc || echo 'export QTDIR=/usr/lib64/qt5' >> ~/.bashrc

source ~/.bashrc
sudo ln -s /bin/tar /usr/bin/ta

\curl -sSL https://get.rvm.io | bash
source /home/user/.rvm/scripts/rvm
rvm install 2.5.8
rvm --default use 2.5.8
```

Установите RhoMobile gems:

```
gem install rhodes-7.2.gem
gem install rhoconnect-client-7.2.gem
```

Пример разработки на Rhodes

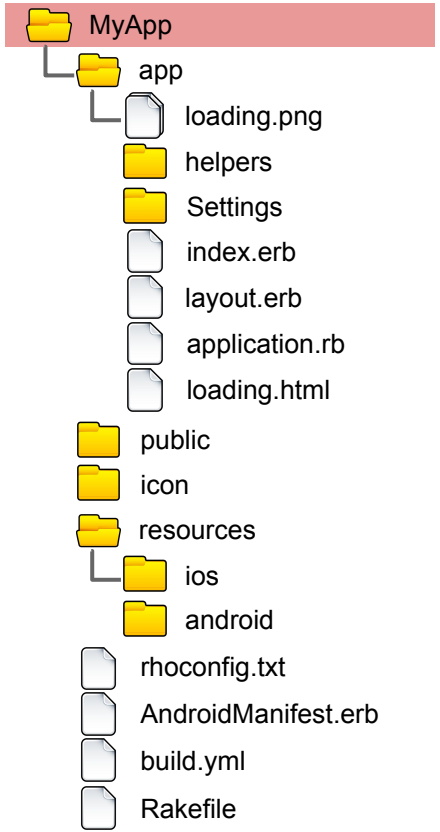
Создадим простое приложение

Сгенерируем наше простое приложение:
(подробности: http://docs.tau-platform.com/en/7.2/guide/creating_a_project) :

```
$ rhodes app MyApp
```

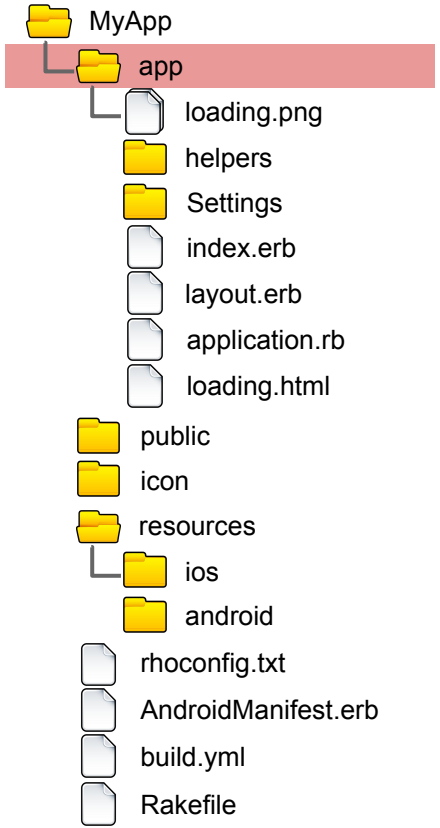
rhodes - утилита командной строки для генерации :
приложений, моделей, расширений.
Генерируемый код полностью работоспособный и можно сразу же собрать и запустить.

Посмотрим внутрь папки приложения



Мы получили MyApp папку где содержится код приложения, ресурсы, настроечные файлы и т.п.

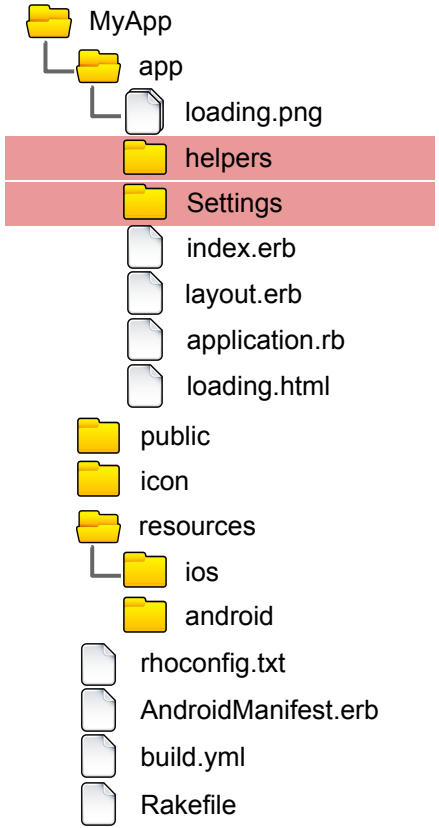
Посмотрим внутрь папки приложения



“app” содержит код приложения - ***.rb** и ***.erb** (шаблоны) файлы

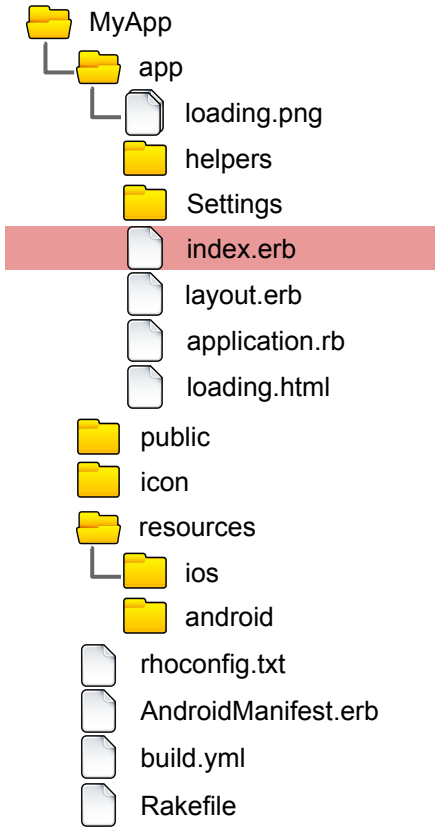
Во время работы приложения эта папка непосредственно находится в корне локального HTTP сервера.

Посмотрим внутрь папки приложения



Папки содержат вспомогательный Ruby код

Посмотрим внутрь папки приложения

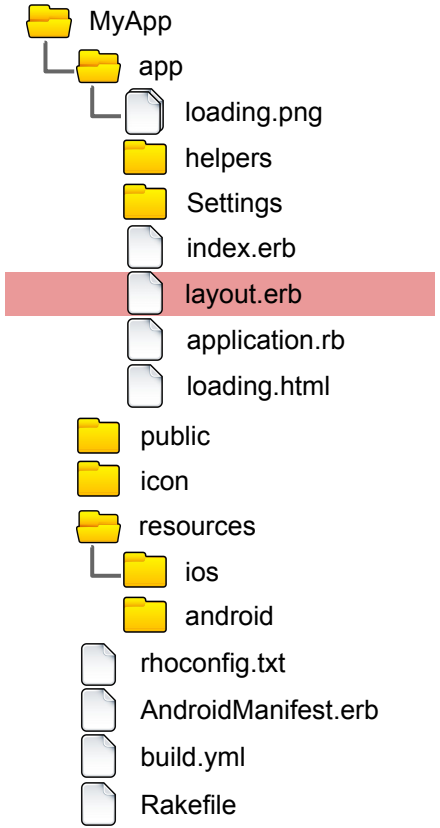


erb шаблон для index страницы (стартовая страница в приложении по умолчанию)
Два шага генерации HTML :
Шаблон страницы содержит только **content**, а общая часть с загрузкой необходимых стилей CSS или JS файлов находится в отдельном шаблоне - **layout**

```
<div class="container-fluid">



  <div class="row">
    <div class="list-group">
      <a href="#" class="list-group-item">
        <span class="glyphicon glyphicon-chevron-right pull-right" aria-hidden="true">
</span>
        Add link here...
      </a>
    </div>
  </div>

</div>
```

Посмотрим внутрь папки приложения

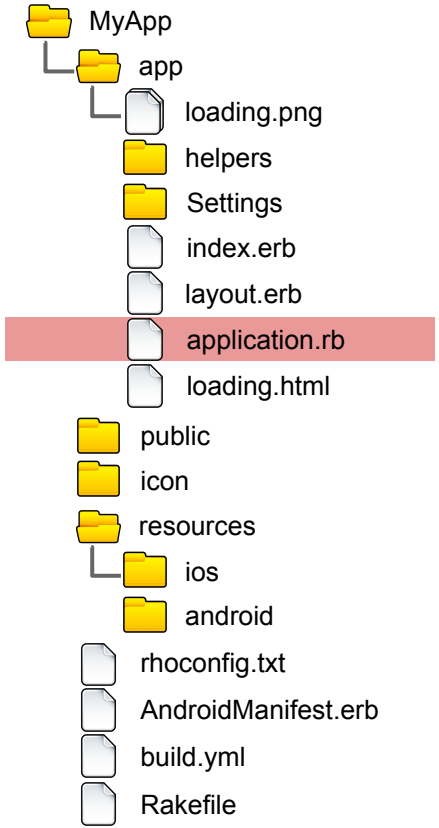
erb шаблон для всех страниц (может быть перекрыт в каждом контроллере)




```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
  <title>MyApp</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0,
maximum-scale=1.0, user-scalable=0"/>

loading of CSS and JS ...

</head>

<body data-platform="<%= Rho::System.getProperty('platform') %>">
<%= @content %>
</body>

</html>
```

фреймворк помещает сгенерированный код страницы сюда

Посмотрим внутрь папки приложения

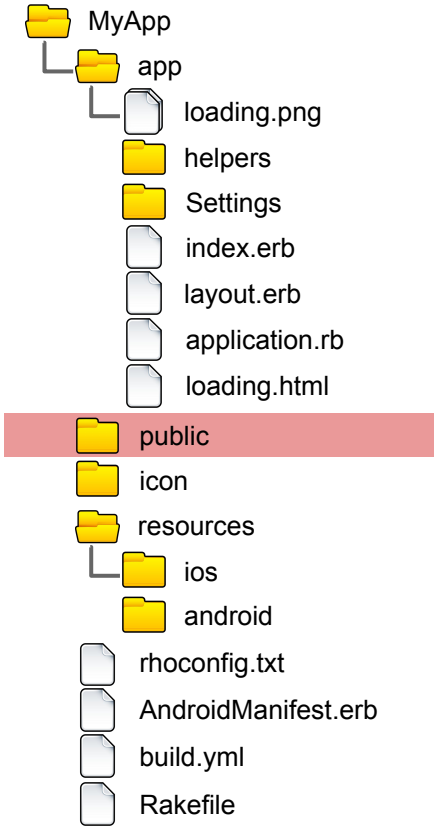


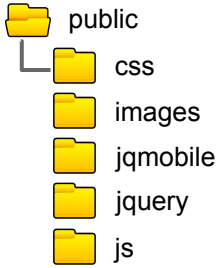
Класс с кодом обработчиков событий приложения - активация, деактивация и т.п.

```
require 'rho/rhoapplication'

class AppApplication < Rho::RhoApplication

  def initialize
    # Tab items are loaded left->right, @tabs[0] is leftmost tab in the tab-bar
    # Super must be called *after* settings @tabs!
    @tabs = nil
    #To remove default toolbar uncomment next line:
    #@@toolbar = nil
    super

  end


end
```

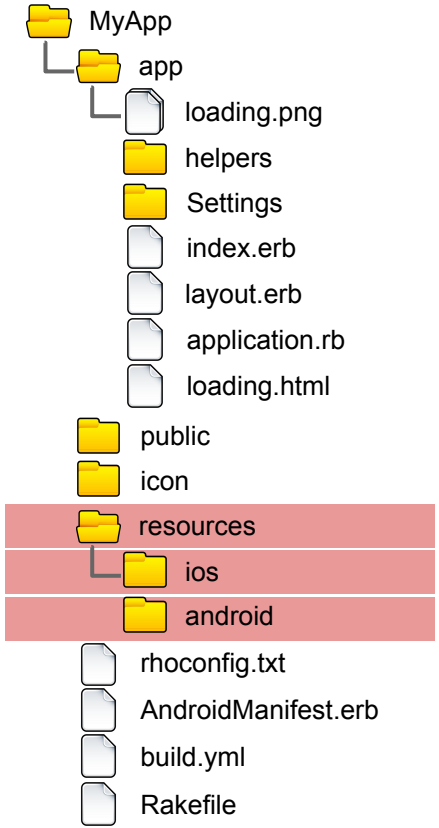
Посмотрим внутрь папки приложения

папка содержит статические файлы локального HTTP сервера: CSS, JS, ресурсы и т.п.





Посмотрим внутрь папки приложения

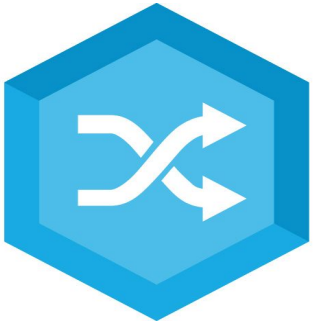


Папка содержит ресурсы необходимые для создания приложения для платформ - иконки, изображения, иные ресурсы.

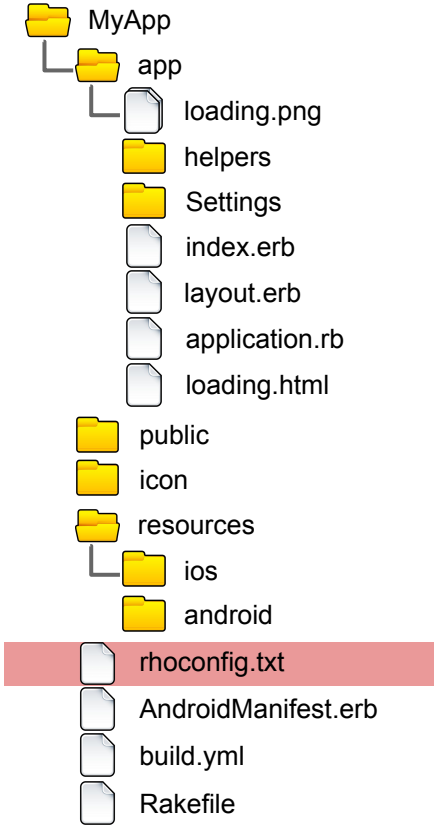
Подробнее в документации:

http://docs.tau-platform.com/en/7.3/guide/app_icon_splash

http://docs.tau-platform.com/en/7.3/guide/build_ios



Посмотрим внутрь папки приложения

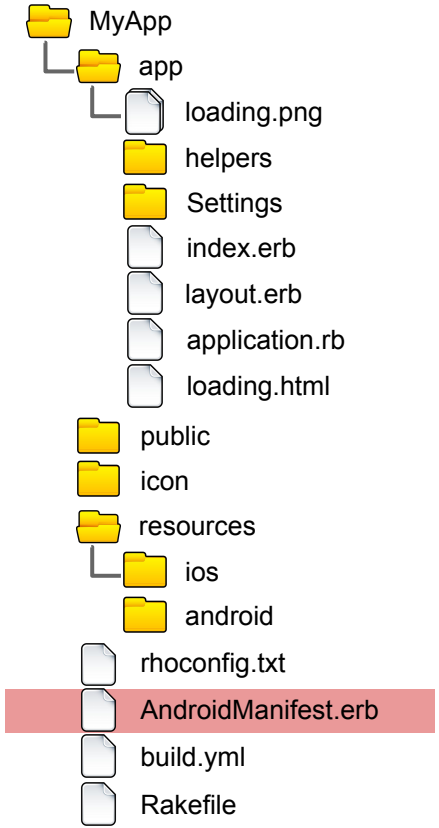


Конфигурационный файл приложения - используется во время работы приложения

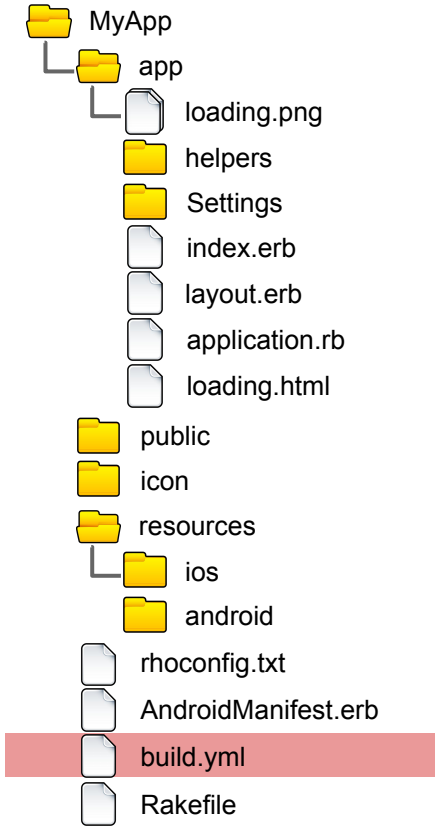
```
# startup page for your application
start_path = '/app'

options_path = '/app/Settings'

# Rhodes log properties
MinSeverity  = 1
LogToOutput = 1
MaxLogFileSize=50000
logserver = 'http://rhologs.heroku.com'
logname='MyApp'

syncserver = ''
sync_poll_interval=0

...
```

Посмотрим внутрь папки приложения

Шаблон манифеста для Android. Подробнее в документации:
http://docs.tau-platform.com/en/7.3/guide/build_android



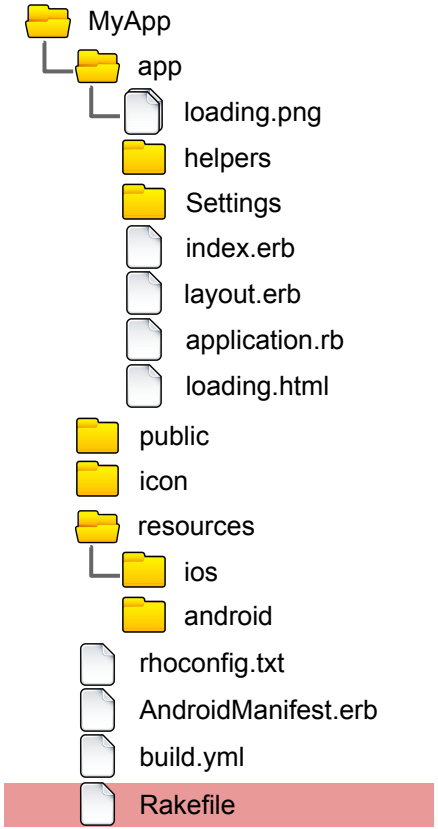
Посмотрим внутрь папки приложения



Конфигурация сборки - настройки используемые для сборки приложения, в том числе настройки доступа к разным возможностям на конкретной платформе и настройка специфичных параметров

```
name: MyApp
version: 1.0
vendor: rhomobile
build: debug
applog: rholog.txt

capabilities:
  - camera

iphone:
  configuration: Release
  sdk: latest
  BundleIdentifier: com.rhomobile.myapp
  BundleURLScheme: myapp

android:
  version: 4.1.0
  logcatFilter: APP:I StrictMode:I DEBUG:I *:E

extensions: []
```

Посмотрим внутрь папки приложения



Стандартный Ruby скрипт для запуска rake (Ruby решение для сборки и конфигурирования сборки - используется в RhoMobile для сборки приложения) команд (build, run, etc.)

Добавим в приложение базу данных и модель для нее

Давайте добавим в наше приложение базу данных и модель для работы с ней - Rhodes генератор добавит все необходимые файлы включая шаблоны страниц для работы с базой и необходимый код контроллера - все это просто образец - разработчик может просто создать это сам или менять созданный генератором код.
(подробнее: http://docs.tau-platform.com/en/7.3/guide/rhom_ruby)

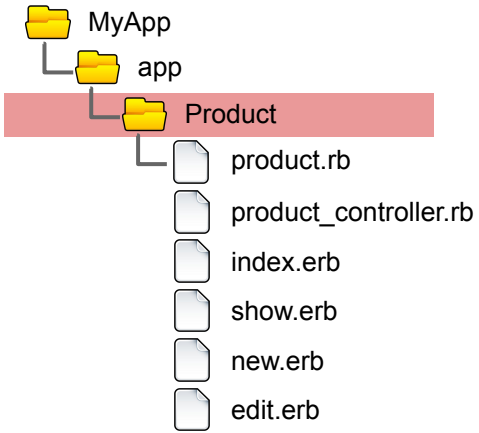
запустите Rhodes генератор из папки приложения:

```
$ rhodes model Product name,brand,price
```

В данном случае мы добавляем модель Product с тремя полями name,brand,price

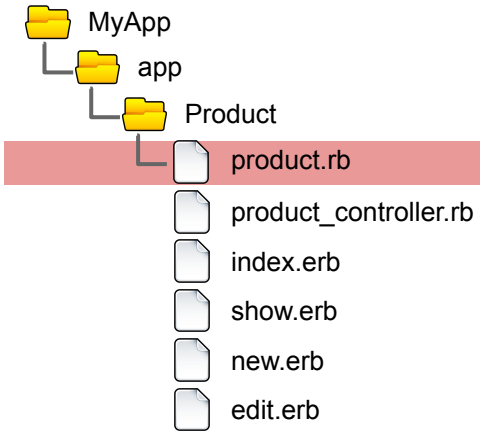
Рассмотрим сгенерированный код

В папке **app** добавился новый каталог: **Product** внутри которого несколько файлов



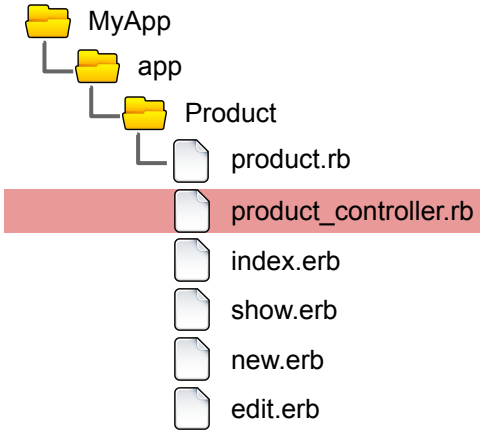
Рассмотрим сгенерированный код

В этом файле определена модель. PropertyBag схема используется по умолчанию.




```
# The model has already been created by the framework, and extends Rhom::RhomObject
# You can add more methods here
class Product
  include Rhom::PropertyBag

  # Uncomment the following line to enable sync with Product.
  # enable :sync

  #add model specific code here
end
```

Рассмотрим сгенерированный код



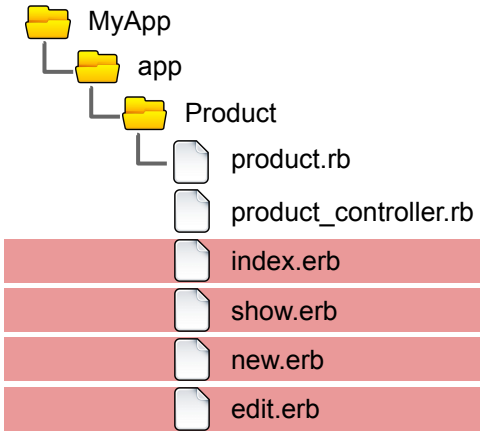
product_controller.rb - Контроллер с ruby кодом.
методы контроллера вызываются из WebView с URL типа /app/Controller/**method** и возвращают HTML страницу созданную из шаблона страницы **method**.erb по умолчанию (может быть перекрыто в коде контроллера вплоть до полного определения возвращаемого контента).
Если method не задан то просто создается страница на базе шаблона и возвращается.

```
require 'rho/rhocontroller'
require 'helpers/browser_helper'

class ProductController < Rho::RhoController
  include BrowserHelper

  def index
    @products = Product.find(:all)
    render :back => '/app'
  end

  def edit
    @product = Product.find(@params['id'])
    if @product
      render :action => :edit, :back => url_for(:action => :index)
    else
      redirect :action => :index
    end
  end

...
```

берет все обьекты модели **Product**

@params - hash содержащий все параметры вызова (query часть URL)

Рассмотрим сгенерированный код



Набор **views** - **erb** шаблоны для генерации базовых страниц для работы с моделью.
index.erb страница по умолчанию. Код метода **index** контроллера также вызывается, когда локальный сервер получает запрос типа **/app/Controller/**

```
<div class="container-fluid">

  <div class="row page-title">
…
  </div>

  <div class="row">
    <div class="list-group">
      <% @products.each do |product| %>

        <a href="<%= url_for :action => :show, :id => product.object %>" class="list-group-item">
          <span class="glyphicon glyphicon-chevron-right pull-right" aria-hidden="true"></span>
          <%= product.name %>
        </a>

      <% end %>
    </div>
  </div>

</div>
```

перебор всех обьектов модели и генерация ссылок чтобы открыть/редактировать каждый обьект

Финальная настройка

Последнее изменение - меняем наш стартовый URL в **rhoconfig.txt** на адрес страницы модели Product, чтобы на старте приложения мы сразу оказались на странице работы с этой моделью :

```
# startup page for your application
start_path = '/app/Product'
```

Запуск/сборка приложения

Что такое rake

Для сборки, запуска и прочих действий с приложением, RhoMobile использует утилиту rake. Поэтому в папке приложения находится файл Rakefile - это скрипт-точка входа в основной скрипт RhoMobile. Сама утилита в виде пакета для ruby (gem) обычно уже установлена вместе с ruby, чтобы поставить вручную надо выполнить(необходим ruby):
gem install rake
Для запуска команд используется синтаксис:
rake command[parameters]
Например (запуск приложения на iOS симуляторе):
rake run:iphone
или (запуск приложения на присоединенном Android устройстве):
rake run:android:device

Rake - инструмент для автоматизации сборки программного кода, написанный на Ruby, и применяющийся в основном для проектов на Ruby (но используется также и для проектов на других языках). Подобен SCons, Make и Apache Ant, но имеет несколько отличий, в частности, так называемые Rakefile (аналоги Makefile в утилите make) используют синтаксис Ruby
Сайт - https://github.com/ruby/rake

Запуск на iOS

запуск на iPhone Simulator:

```
$ rake run:iphone
```

Также можно сгенерировать XCode проект и работать с ним из XCode.
Создание XCode проекта :

```
$ rake rake build:iphone:setup_xcode_project
```

Созданный XCode проект :

Подробнее: http://docs.tau-platform.com/en/7.3/guide/build_ios

Запуск на Android

Запуск на Android Emulator:

```
$ rake run:android
```

Сборка и запуск приложения на подсоединенном через USB кабель Android устройстве :

```
$ rake run:android:device
```

Подробности: http://docs.tau-platform.com/en/7.3/guide/build_android

Запуск на Аврора ОС

Настраиваем в Sailfish IDE (Qt creator) новое устройство, вписывая ip и ssh пароль.
Напоминаем что папки rhodes, rhoconnect-client и приложения находится в домашней директории !
Чтобы настроить пути в приложении, отредактируйте конфигурационный файл приложения - **build.yml**
путь к rhodes:
sdk: /path/rhodes
путь к rhoconnect-client:
paths:
extensions: ["path/rhoconnect-client/ext"]

Запуск приложения на устройстве:

```
$ rake run:sailfish:device
```

Создание установочного пакета

```
$ rake device:sailfish:production
```

Подробнее: http://docs.tau-platform.com/en/7.3/guide/build_sailfish

Запуск на Ubuntu, Alt Linux, Astra Linux, Red OS, Rosa Linux

Запуск приложения:

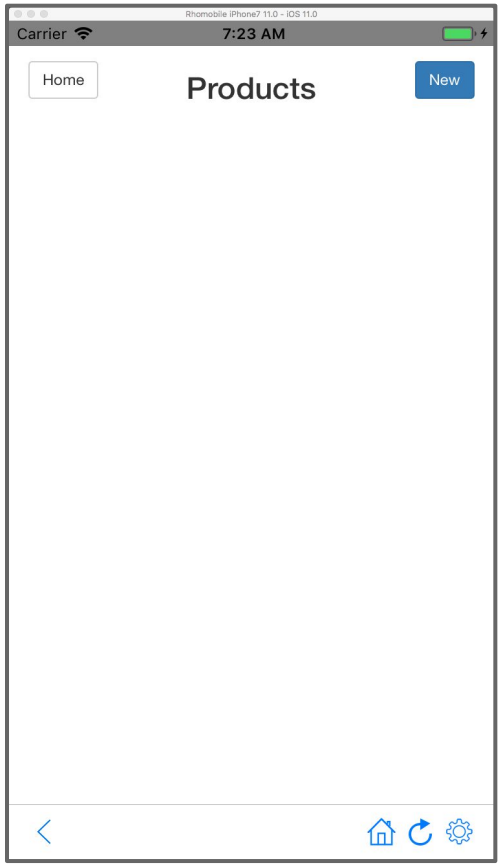
```
$ rake run:linux
```

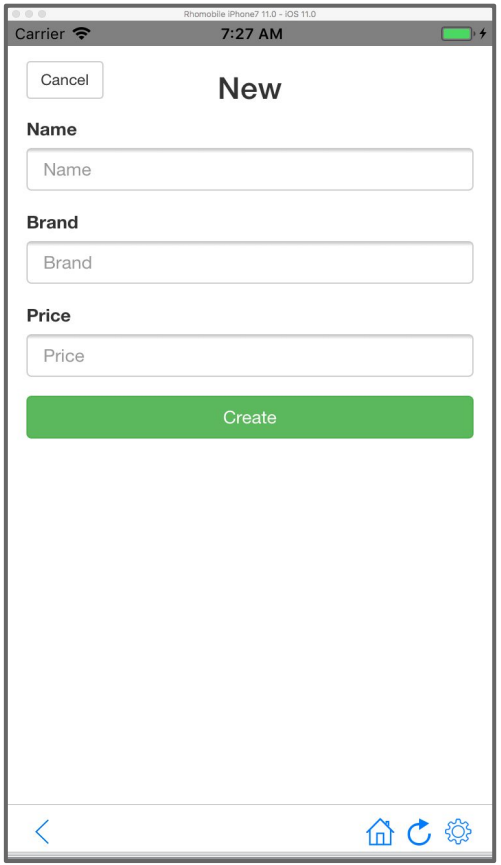
Создание установочного пакета приложения (кроме Rosa Linux)

```
$ rake device:linux:production
```

Результат

Скриншоты приложения:





Дополнительно:

Стандартное APi :
http://docs.tau-platform.com/en/7.3/guide/apisummary

Работа с удаленным сервером через RestAPI из Ruby и JS кода в Rhodes :
http://docs.tau-platform.com/en/7.3/guide/web_services

Как подключать сторонние GEM-ы в Rhodes приложение :
http://docs.tau-platform.com/en/7.3/guide/ruby_extensions

Как создавать нативные расширения для Rhodes приложения :
http://docs.tau-platform.com/en/7.3/guide/native_extensions

Как легко и просто синхронизировать локальную базу данных с базой на удаленном сервере при помощи решения RhoConnect:
http://docs.tau-platform.com/en/7.3/guide/synchronization

Дополнительные примеры приложений :
https://github.com/tauplatform/rhodes-system-api-samples
https://github.com/tauplatform/kitchensinkRuby
https://github.com/tauplatform/inventoryDemo-mobileApp
https://github.com/tauplatform/kitchensinkJS
https://github.com/tauplatform/universal-push-example

Контакты

На нашем сайте размещена подробная информация, документация, форум и т.п.

http://tau-platform.com