

# **XII. Архитектури и технологии за разработване на мобилни приложения**



## Видове архитектури

- Към настоящия момент съществуват три основни вида архитектури за мобилни приложения – тънък клиент, дебел клиент и смарт клиент.
- За да се определи коя от тези архитектури е най-подходяща за разработване на дадено мобилно приложение трябва да се отговори на следните въпроси:
  - Кой са крайните потребители на приложението?
  - Потребителите имат ли специални изисквания към устройството или приложението ще определя неговия вид?
  - Каква е основната цел на приложението?



## Видове архитектури

- Необходим ли е постоянен достъп до бази от данни?
- Приложението изисква ли безжична връзка и от какъв тип?
- По какъв начин ще се използва приложението?
- По какъв начин ще се разпространява и обновява приложението?
- От архитектурата на мобилното приложение зависи изборът на средства за неговото разработване.



*Видове мобилни приложения*

## Native

Advanced UI interactions  
Fastest performance  
App store distribution

full  
capability

## Hybrid

Web developer skills  
Access to native platform  
App store distribution

single  
platform

multiple  
platforms

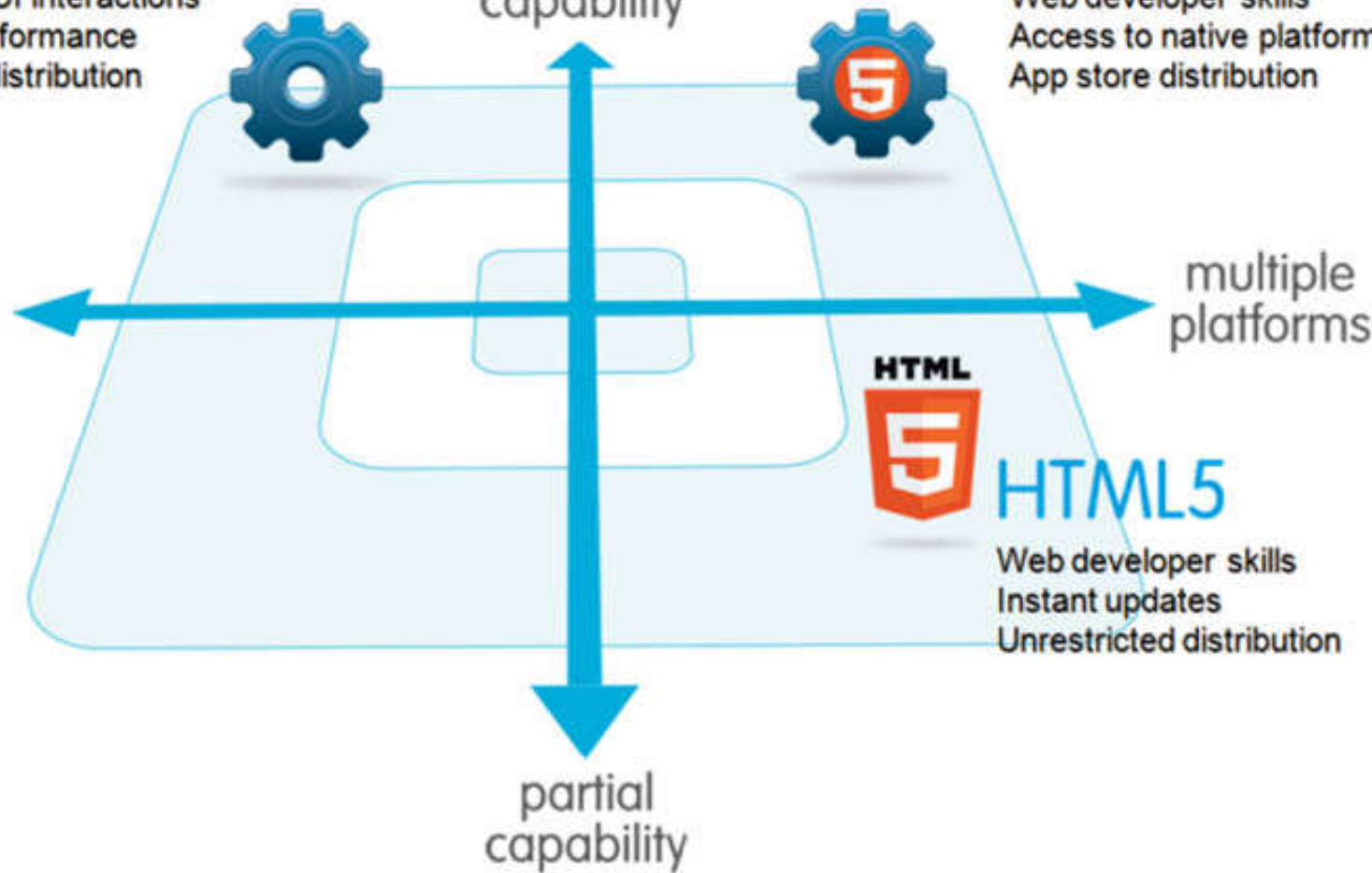
HTML



HTML5

Web developer skills  
Instant updates  
Unrestricted distribution

partial  
capability



# Видове архитектури

## 1. Тънък клиент

- Под термина „тънък клиент“ се разбира Интернет базирано приложение, което се изпълнява от мобилния браузър на клиентското устройство.
- Това означава, че на клиентското устройство не е необходимо инсталиране на софтуер, освен web-браузър.
- Тънкият клиент трябва да бъде напълно независим от специфичен браузър или операционна система.

# YOUR APP

## WEB UI & LOGIC



### Mobile UI Components

Develop mobile web applications in the familiar force.com environment on a proven architecture stack



### Visualforce

Develop mobile web applications in the familiar force.com environment on a proven architecture stack



### JavaScript Remoting

Invoke Apex controller methods directly from a mobile application for optimized performance



### Static Resources

Include mobile optimized third parties frameworks like JQuery Mobile, iScroll, and Sencha



# Видове архитектури

## 1. Тънък клиент. Предимства и недостатъци

- Този тип мобилни приложения имат следните предимства:
  - по-бързо разработване в сравнение с приложенията от типа „дебел“ клиент;
  - не е необходимо инсталиране на софтуер в паметта на потребителското мобилно устройство;
  - използването на мобилен web-браузър за възпроизвеждане на приложението елиминира потенциалните проблеми от несъвместимост с някои операционни системи или приложен софтуер, като осигурява възможност за разработване на едно приложение;



# Видове архитектури

## 1. Тънък клиент. Предимства и недостатъци

- ▶ достъп от всяко място и по всяко време, което осигурява по-широко разпространение;
- ▶ по-ниска цена за поддържане на приложението;
- ▶ възможност за лесно обновяване на съдържанието;
- ▶ Недостатъците на приложенията от типа тънък клиент са:
  - ▶ по-ниска степен на интерактивност в сравнение с дебелите клиенти, тъй като не всички мобилни web-браузъри поддържат в еднаква степен Java Script;
  - ▶ ограничения в големината на медийните файлове, които се пренасят през безжичния Интернет;



# Видове архитектури

## 1. Тънък клиент. Предимства и недостатъци

- ▶ поради по-ниската степен на интерактивност, тези приложения изискват информацията да се представя на малки порции, на повече екрани, което води до изпълнение на повече действия от потребителя, заявки към сървъра и по-бавно изпълнение в сравнение с дебелия клиент;
- ▶ поради непрекъснатата поява на нови мобилни web-браузъри и бързата смяна на технологията е трудно тънките клиенти да се възползват от най-добрите им възможности. За да осигурят съвместимост, разработчиците са принудени да се съобразяват с възможностите на по-старите браузъри;



# Видове архитектури

## 1. Тънък клиент. Предимства и недостатъци

- ▶ потребителите не могат да разработят offline с приложението и поради тази причина се изисква непрекъснатата безжична Интернет връзка;
- ▶ все по-голямо разнообразие от мобилни web-браузъри, притежаващи различни функционални възможности;
- ▶ евентуална необходимост от идентификация типа на мобилното устройство и от адаптиране на приложението;
- ▶ ограничен достъп до специфичните ресурси на мобилното устройство поради съображения за сигурност.



# Видове архитектури

## 2. Дебел клиент

- Под термина „дебел клиент“ се разбира приложение, което се зарежда в паметта на потребителското мобилно устройство.
- Може да комуникира със сървър посредством безжична или кабелна връзка с цел синхронизиране на данни.



# YOUR APP

## NATIVE UI & LOGIC



### OAuth2

Secure authentication and refresh token management



### API Wrappers

Interact with Salesforce REST APIs with popular mobile platform languages



### Secure Offline Storage

Store business data on a device with enterprise-class security



### Push Notifications

Dispatch real-time alerts directly to mobile devices



# Видове архитектури

## 2. Дебел клиент. Предимства и недостатъци

- Този тип мобилни приложения имат следните предимства:
  - притежават повече възможности за възпроизвеждане на медийни файлове, по-добър потребителски интерфейс и по-голяма степен на интерактивност в сравнение с тънките клиенти;
  - могат да съхраняват сравнително големи обеми от данни директно на мобилното устройство, което ускорява достъпа до тях;
  - изпълняват се по-бързо и не изискват непрекъснатата връзка със сървър;



# Видове архитектури

## 2. Дебел клиент. Предимства и недостатъци

- ▶ могат директно да взаимодействат с ресурсите на мобилното устройство, като получават полезна информация – местоположение, списък с контакти, информация за ускорение и др.;
- ▶ средите за разработване се подобряват много бързо;
- ▶ може да бъде получавана информация, по какъв начин потребителите взаимодействат с приложението.

# Видове архитектури

## 2. Дебел клиент. Предимства и недостатъци

- ▶ Недостатъци на дебелия клиент са:
  - ▶ по-големи разходи за оборудване и за разработване, ако приложението трябва да поддържа различни платформи;
  - ▶ трудно обновяване, добавяне на нови функционални възможности и по-дълго време за отстраняване на грешки, тъй като се изисква инсталиране на софтуера в паметта на мобилното устройство;
  - ▶ в момента съществуват много среди за разработване и трябва да се подбере тази, която най-добре отговаря на нуждите;





# Видове архитектури

## 2. Дебел клиент. Предимства и недостатъци

- ▶ разработването на богати приложения изисква повече средства, отколкото разработването на мобилен web сайт;
- ▶ трудна преносимост на приложението на друг тип мобилно устройство и/или друга операционна система, което най-често води до необходимост от разработване на различна версия за всяка мобилна платформа.

# Видове архитектури

## 3. Смарт клиент

- Под този термин се разбира приложение, което използва локални ресурси на мобилно устройство, базира се на web услуги и може да се разпространява и обновява от централизиран сървър.
- Тези приложения могат да работят, както безжично свързани към сървър, така и без такава връзка.
- Комбинират възможностите на тънките и дебелите клиенти.



## Visualforce

Develop mobile web apps in the familiar Force.com environment on a proven architecture stack



## JavaScript Remoting

Invoke Apex controller methods directly from a mobile application for optimized performance



## Mobile Components

Reusable Visualforce-based building blocks for constructing mobile apps through customizable components

## Container

Embed HTML5 apps inside a container to access powerful native device functionality



# Видове архитектури

## 3. Смарт клиент. Предимства и недостатъци

- ▶ Предимствата на смарт клиентите са:
  - ▶ възможност за разпространение и обновяване от сървър посредством Интернет;
  - ▶ по-малко са зависими от платформата, на която работят, тъй като се базират на web услуги;
  - ▶ по-голяма интерактивност и по-бързо изпълнение на задачите, както при дебелите клиенти;
  - ▶ някои от платформите притежават вградена сигурност и възможности за управление;
  - ▶ някои от платформите осигуряват използване на вградени ресурси за комуникация (e-mail, чат и др.).

# Видове архитектури

## 3. Смарт клиент. Предимства и недостатъци

- ▶ Недостатъци на смарт клиентите са:
  - ▶ липсва стандартизация на използваните API;
  - ▶ по-висока цена, тъй като за разлика от платформите за разработване на „дебели“ клиенти, тези за разработване на приложения от типа „смарт клиент“ не са безплатни;
  - ▶ варират в своята „дебелина“ в зависимост от функционалните възможности на мобилното устройство.



## Съвременни технологии за бързо разработване

- Мобилните платформи са най-бързо развиващите се, и в резултат на това, много хора и фирми се обръщат именно към мобилните приложения, за да ангажират по-ефективно своята аудитория.
- Това обаче не е лесна задача, тъй като създаването на мобилни приложения изисква добро владение на съвременните езици за програмиране, като например Objective C (iOS) или Java (Android).



## Съвременни технологии за бързо разработване

- Съществува набор от инструменти, които позволяват проектирането на мобилни приложения с готови интерфейси или уеб технологии, а след това 'превеждат' програмния код, така че да бъде 'разбран' от различни устройства.
- Такъв подход позволява бързото създаване на приложения, които да са съвместими с голям брой устройства и да предлагат богат набор от функционалност.



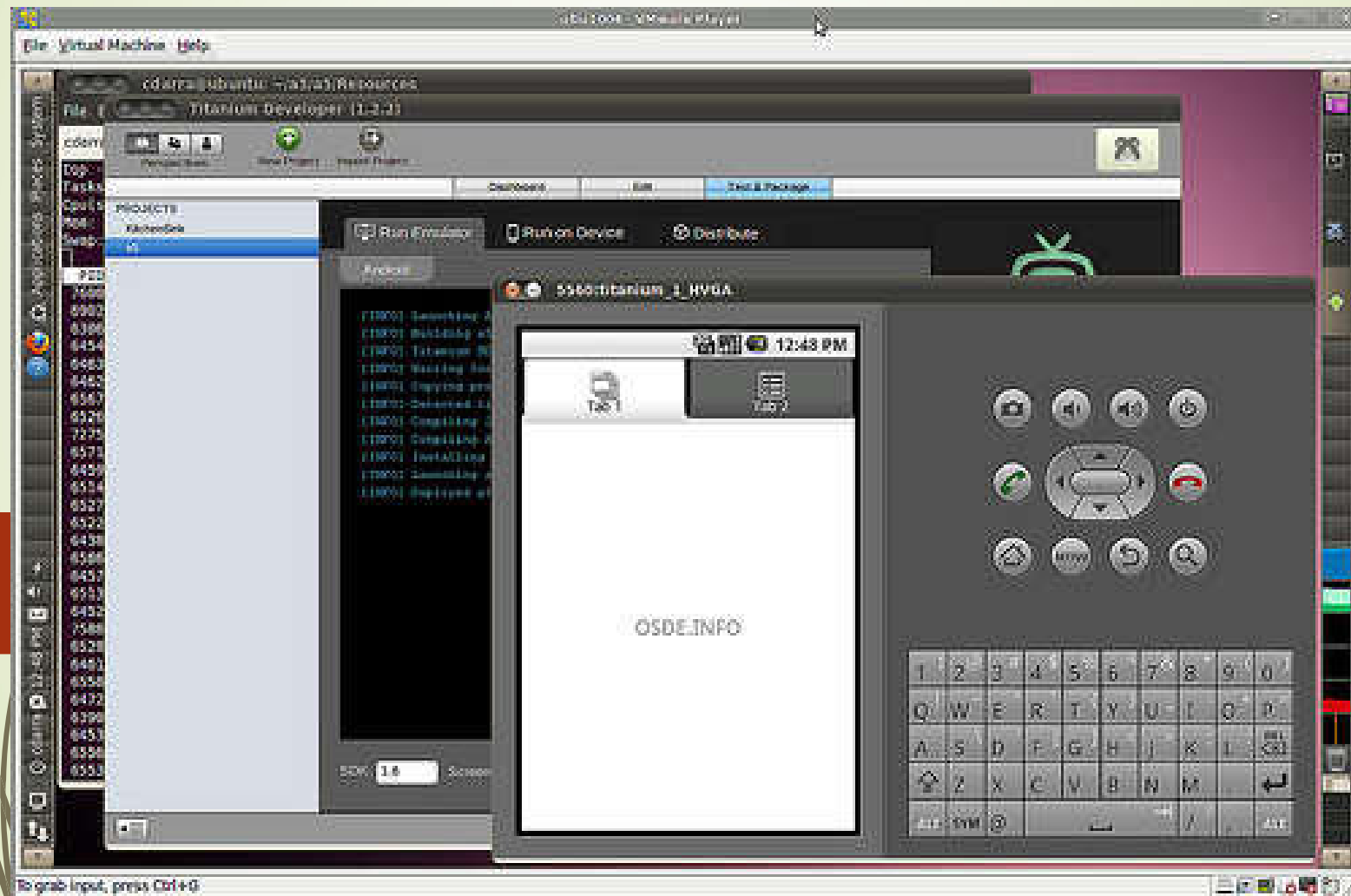
# Съвременни технологии за бързо разработване

## Appcelerator Titanium

- Appcelerator Titanium SDK е един от водещите софтуерни инструменти за разработване на мобилни приложения. Той има 1.5 милиона активни програмисти и съществуват около 22 000 разработени приложения.
- Библиотеките с програмен код и помощните платформени инструменти позволяват на програмистите да се фокусират върху създаването на приложения, които могат да бъдат инсталирани на различни мобилни устройства и настолни компютри, а самите приложения могат да бъдат кодирани с езици като PHP, Python, CCS, HTML, и Javascript.



# Appcelerator Titanium SDK





# Съвременни технологии за бързо разработване

## MIT App Inventor

- MIT App Inventor представлява облачна среда за бърза визуална разработка на приложения за операционната система Android.
- App Inventor използва визуален интерфейс за проектиране, като логиката на работа на програмата може да се създава без да се изисква познания по програмиране на Java и Android SDK.
- Логическите блокове се свързват един с друг, подобно на подреждането на пъзел.
- App Inventor проектът се създава чрез блокове, които се наричат компоненти.



# Съвременни технологии за бързо разработване

## MIT App Inventor

- Такива са например Label за извеждане на текст, TextBox за въвеждане на информация от потребителя, компонент Camera и др.
- В полето Palette можете да видите различни категории компоненти – от основни компоненти, например за въвеждане и извеждане на текст, до много по-специализирани компоненти за възпроизвеждане на медия и анимация, както и компоненти, които служат като интерфейс за управление сензорите на устройството.
- Компонентите имат присвоени поведение, методи и свойства. Някои от свойствата могат да бъдат променяни, а други могат само да се четат.

# App Inventor (prozorec za dizajn)

The screenshot displays the App Inventor web interface for a mobile application named "PictureSensorMove". The interface is divided into several panels:

- Palette:** A sidebar on the left containing various components categorized into User Interface, Layout, Media, Drawing and Animation, Sensors (AccelerometerSensor, BarcodeScanner, Clock, LocationSensor, NearField, OrientationSensor, ProximitySensor), Social, Storage, Connectivity, and LEGO® MINDSTORMS®.
- Viewer:** The central workspace showing a mobile app preview. It includes a header with "Screen1", "Add Screen...", and "Remove Screen" buttons. The preview area shows a mobile screen with a NASA nebula image, a small rocket icon, and a status bar at the top with signal, Wi-Fi, and battery icons, and a time of 9:48. Below the preview is a "Non-visible components" section showing "Clock1" and "OrientationSensor1".
- Components:** A panel on the right showing a tree view of the app's components: "Screen1" (containing "Canvas1", "ImageSprite1", "Clock1", and "OrientationSensor1").
- Properties:** A panel on the right showing the properties for the selected "ImageSprite1" component. Properties include: Enabled (checked), Heading (0), Height (Automatic...), Width (Automatic...), Interval (100), Picture (40.png...), Rotates (checked), Speed (0.0), Visible (checked), X (143), Y (151), and Z (1.0).
- Media:** A section at the bottom right showing a list of media files: "images.jpeg" and "40.png", with an "Upload File ..." button.

# App Inventor (редактор на блокове)

The screenshot displays the App Inventor for Android Blocks Editor interface for a project named "BballQuiz". The interface includes a top toolbar with buttons for "Built-In", "My Blocks", "Saved", "Undo", "Redo", and "Connect to phone". On the left, there is a "My Blocks" palette with categories like "My Definitions", "AnswerButton", "AnswerPromptLabel", "AnswerText", "HorizontalArrangement1", "HorizontalArrangement2", "Image1", "NextButton", "QuestionLabel1", "RightWrongLabel", "RightWrongLabel2", "Screen1", and "Storage".

The main workspace shows two event-driven code blocks:

- when Screen1.Initialize:** A "do" block containing a "set" block that sets "QuestionLabel1.Text" to the result of a "select list item" block. The "select list item" block is configured with "list" set to "global QuestionList" and "index" set to "number 1".
- when NextButton.Click:** A "do" block containing several steps:
  - An "if" block testing "global currentQuestionIndex = call length of list list global QuestionList".
  - A "then-do" block setting "global currentQuestionIndex" to "number 0".
  - A "set global" block setting "currentQuestionIndex" to "global currentQuestionIndex + number 1".
  - A "set" block setting "QuestionLabel1.Text" to the result of a "select list item" block with "list" set to "global QuestionList" and "index" set to "global currentQuestionIndex".
  - A "set" block setting "Image1.Picture" to the result of a "select list item" block with "list" set to "global PictureList" and "index" set to "global currentQuestionIndex".



# Съвременни технологии за бързо разработване

## AppMaker

- AppMaker позволява създаването на мобилни приложения без да се налага програмиране.
- За целта се използва метода 'посочи-с-мишката-и-кликни', при което се взема готово съдържание от уеб сайт и го вгражда автоматично в желано мобилно приложение.
- Позволява на потребителите да създават Android, iOS и Windows Phone приложения. Средата е безплатна.
- Потребителите няма нужда да имат опит или познания за програмиране.

# AppMakr

Step 1 Step 2 Step 3

### Available in-app functions

Common Functions

- Websites
- MyBlog
- News
- Photos
- Videos
- Contacts
- Directions
- Calendar
- HTML Page
- Forms
- Docs
- LiveChat

Social Feeds

News & Blogs

Photos & Videos

### Appearance

- Backgrounds
- Header
- Icons

### My Very First App

MyBlog  
infinite monkeys

This fan-originated app is not affiliated with or endorsed by any official party. All images and copyrights remain the property of their rightful owners.

Live Preview

App Saved 32 secs ago

BACK NEXT HELP EXIT

### Video Help

MyBlog

Include RSS feed from your favourite blog, or just about any RSS feed. You will need to find the URL (address) for the feed itself, not just the homepage. Look for this symbol  or the word 'RSS' or 'feed' on the source site.

[Click here to watch the Help Video](#)

### Specs / Info

Icon Title: MyBlog

Icon Background

Feed Url: RSS feed link

SAVE



# Съвременни технологии за бързо разработване

## PhoneGap

- PhoneGap позволява на програмистите да кодират приложения на HTML 5 и JavaScript, без те да пречат на функционирането на основните телефонни функции.
- Софтуерът е с отворен код и дава достъп до интерфейси за различни платформи, в това число iOS, Android, Windows Mobile, Blackberry, и Symbian.
- PhoneGap също така дава възможност на разработчиците да използват вградените APIs устройства - като камера и списъка с контакти - директно в JavaScript.



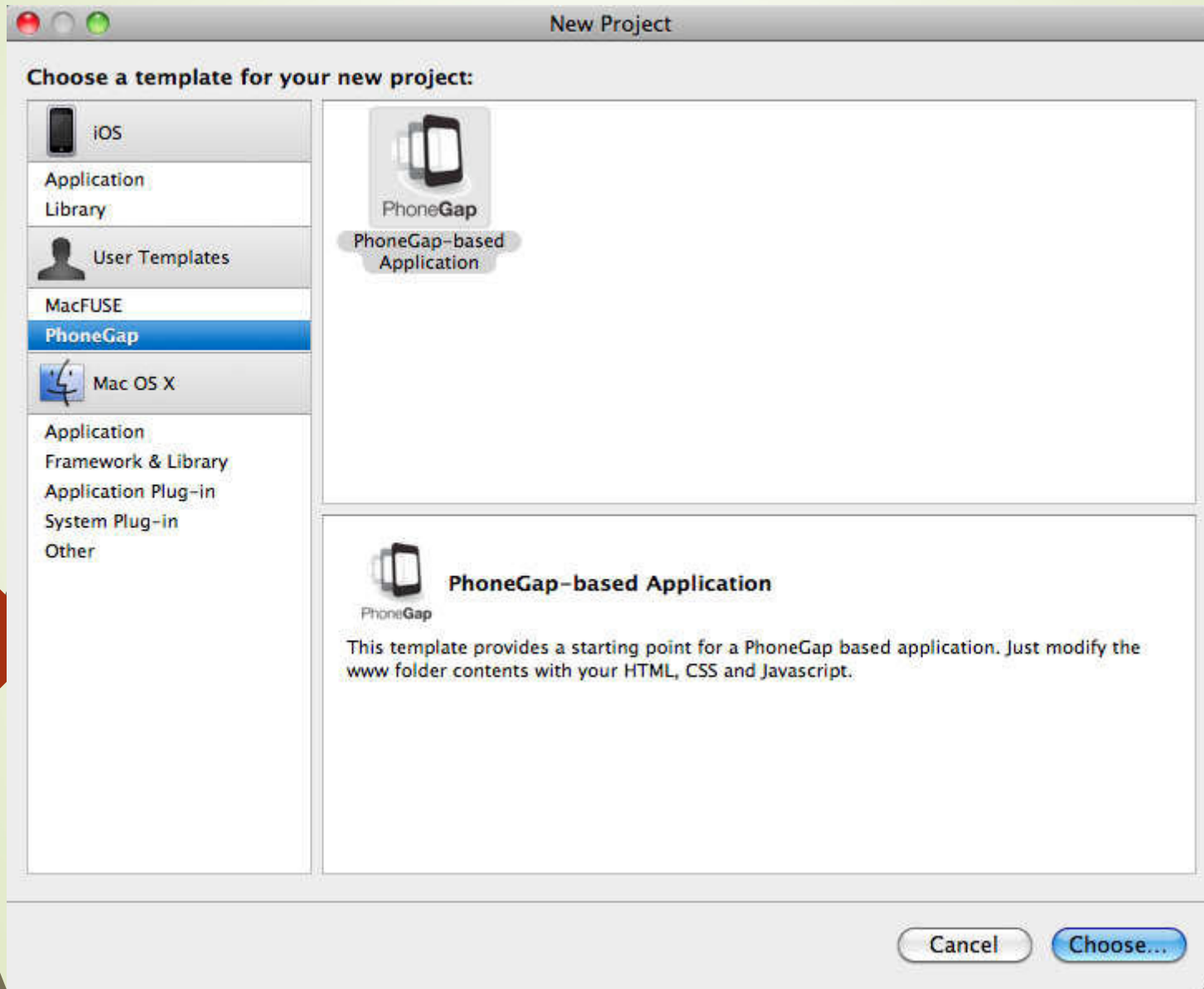


# Съвременни технологии за бързо разработване

## PhoneGap

- PhoneGap също така дава възможност на разработчиците да използват вградените APIs за устройства – като например камера и списъка с контакти - директно в JavaScript.
- Готовите приложения се компилират и се създава инсталационен пакет за всяка платформа.

# PhoneGap





# Съвременни технологии за бързо разработване

## Sencha Touch

- ▶ Sencha предлага различен подход към разработката на мобилни приложения.
- ▶ За разлика от други подобни инструменти, Sencha Touch предлага JavaScript интерфейс за кодиране на уеб-базирани приложения, които не се нуждаят от одобрение за публикуване в App Store.
- ▶ Sencha е инструмент, който изисква само познания по JavaScript и HTML.



# Съвременни технологии за бързо разработване

## Sencha Touch

- Sencha Touch позволява да се разработи уеб базиран софтуер за мобилни устройства, който е платформено независим, функционален като софтуерът разработван за конкретните операционни системи на мобилните устройства и изглежда и работи еднакво добре под операционните системи на iPhone, Android, Windows Phone и др.
- Sencha Touch е първият в света фреймуорк, изградена изцяло със HTML5, CSS3, и Javascript, като предлага мощни контроли за изграждане на потребителски интерфейс, гъвкавост и оптимизация.

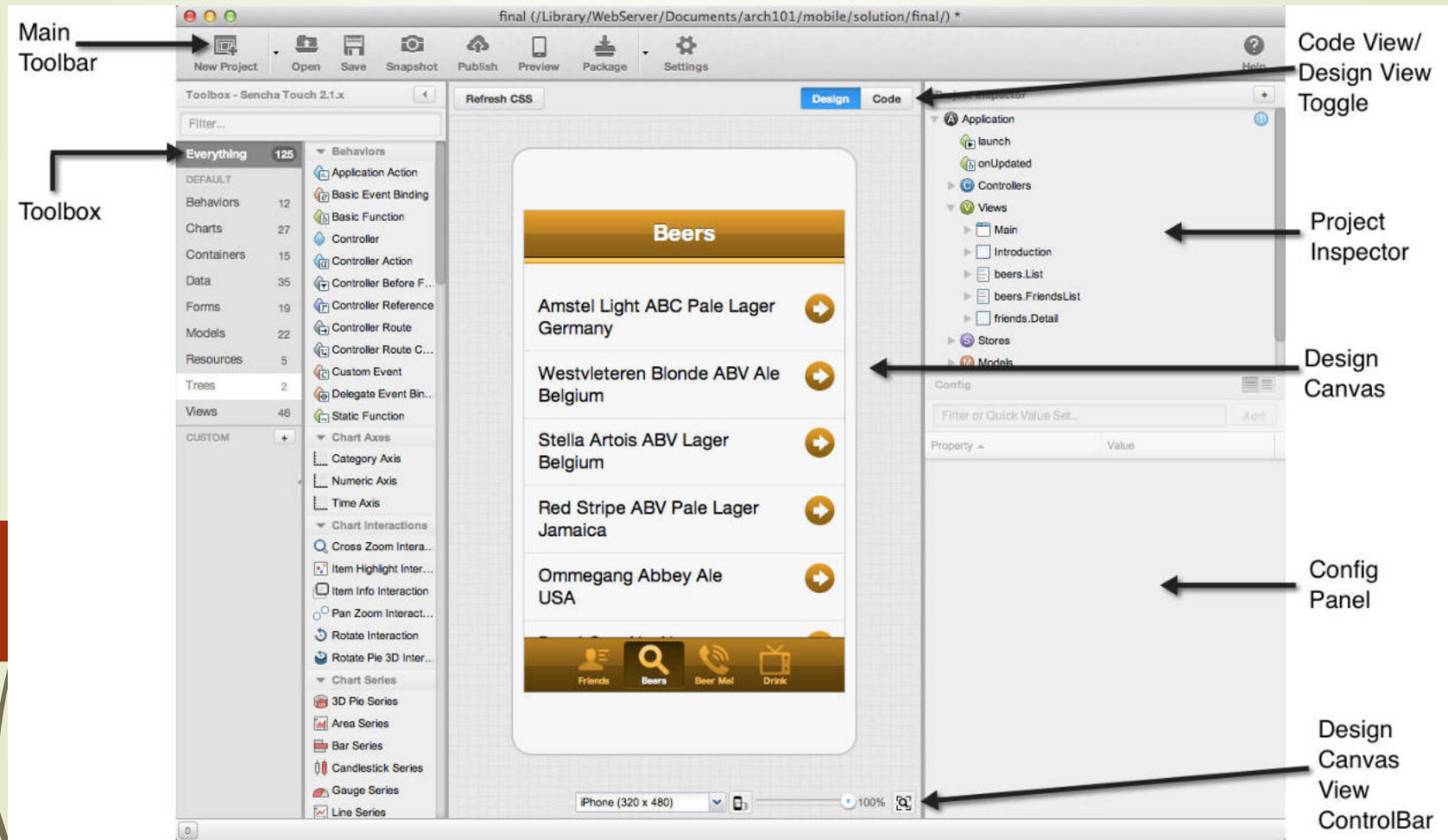


# Съвременни технологии за бързо разработване

## Sencha Touch

- Специфичната употреба на HTML5 предоставя компоненти, като аудио и видео, както и localStorage за записване на данните в офлайн режим.
- Широкото използване на CSS3 в осигурява много добро стилизиране на всички елементи на контролите на потребителския интерфейс без необходимост от използване на изображения.

# Sencha Touch





# Съвременни технологии за бързо разработване

## jQuery Mobile

- jQuery Mobile е оптимизирана работна среда за разработване на мобилни приложения за смартфони и планшети.
- Тя осигурява потребителски интерфейс и навигация за изпълнение на всички популярни мобилни операционни системи. jQuery Mobile е оптимизирана за HTML5 и CSS3.
- Платформата се предлага със следните компоненти: диалогови страници, тулбар, бутони, форматирно съдържание, формуляри, лист изгледи и др. jQuery Mobile включва и много теми. Разработката е съвместима с всички мобилни браузъри.



# Съвременни технологии за бързо разработване

## jQuery Mobile

- Това дава възможност да се разработват приложения с абсолютно идентични интерфейси, без значение от операционната система на мобилното устройство.
- Този понякога е доста трудно да се постигне при разработването на използването на така наречените “дебели клиенти”, написани съответно на Objective C, Java или други специфични за дадена платформа програмни езици.



# jQuery Mobile

Portrait/Landscape orientations

Scale the editing surface to fit to window

jQuery mobile widgets in the Palette

Mobile Navigation view to create and manage jQuery Mobile "page" instances

Property sheets to configure jQuery mobile widgets

Project Explorer

jQueryMobileTest1

- server/java
- server/resources
- adapters
- apps
  - App1
    - android
    - common
      - css
        - App1.css
        - jquery-mobile.css
        - jquery-mobile-threset.css
      - images
      - js

App1.html

Design - jQueryM

App1.html - Eclipse

Device: Apple iPhone 3GS • Skin: con

jQuery Mobile Widgets

- Button
- Checkbox
- Collapsible
- Collapsible Set
- Content
- Control Group
- Dialog
- Field Container
- Footer
- Form
- Grid
- Header

Properties

html > body > Page > Content > Button

jQuery Tag Styles Layout All

Button

Theme: b

Rounded corners

Compact version

Inline

Icon Details

Icon: Right Arrow

Position: Left

mainPage (default)

- worldmap
- local
- facebook
- twitter
- youtube

```
</div>
< a href="#headline" data-role="button" data-icon="arrow-r" data-theme="b" data-corne
< a href="#worldmap" data-role="button" data-icon="arrow-r" data-theme="b" data-mini-
< a href="#us" data-role="button" data-icon="arrow-r" data-theme="b" data-mini="true"
< a href="#local" data-role="button" data-icon="arrow-r" data-theme="b" data-mini="tr
< div data-role="footer" data-position="fixed"
< a data-role="navber"
  href="#facebook"></a> < a
  href="#twitter"></a> < a
  href="#youtube"></a>
  href="#" data-role="button" data-icon="gear" class="ui-btn-right" data-theme="b" data-mini="true" data-corne
</div>
```

Design Source Split