

# Telerik Backend Services

You'll find it empowering ..



**Telerik** Backend Services

So, you're building a modern app

Mobile / Web / Desktop

# Problems you face ..

1. How do you manage & scale Data?
2. Is your data Platform-agnostic?
3. Relational Vs Non-Relational data storage?
4. Do you have consistent APIs on top of your data?
5. How do you handle Users & Authentication?
6. Do you need scalable code running in the cloud?
7. X-Platform Mobile Push Notifications a distant dream?
8. Legacy data holding you back?

Debilitating concerns .. we know

Backend plumbing - done right!



# Why Telerik Backend Services?

-  Complete Backend-as-a-Service (BaaS) offering
-  One portal to rule it all
-  Secure Data storage - relational or non-relational
-  Consistent APIs & Cloud code
-  Social User authentication & Management
-  Easy X-Platform Mobile Push Notifications
-  Easy Data Connectors to On-premise data
-  X-Platform SDKs and extensive product Integrations

That's a lot of features ..

No worries - let's break it down

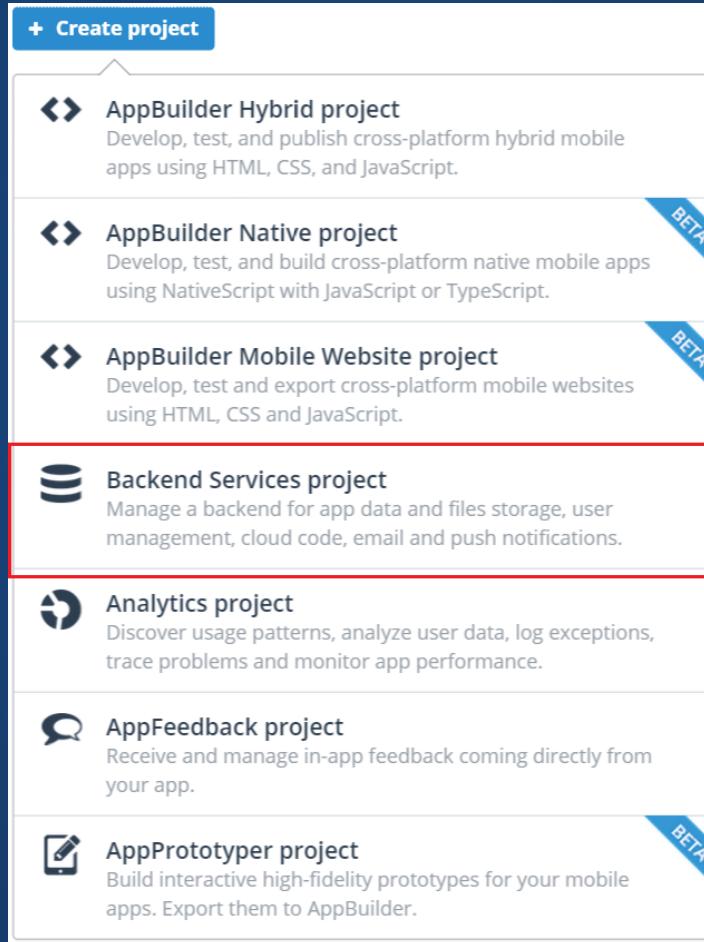
Where do I start?

Simply head to

<https://platform.telerik.com/>

# Sign in & Create an App ..

## Then, start a Backend Services project



The screenshot shows a user interface for creating a new project. At the top, there is a blue button labeled '+ Create project'. Below it, there is a list of project types:

- AppBuilder Hybrid project**: Develop, test, and publish cross-platform hybrid mobile apps using HTML, CSS, and JavaScript.
- AppBuilder Native project**: Develop, test, and build cross-platform native mobile apps using NativeScript with JavaScript or TypeScript. A small blue 'BETA' badge is visible.
- AppBuilder Mobile Website project**: Develop, test and export cross-platform mobile websites using HTML, CSS and JavaScript. A small blue 'BETA' badge is visible.
- Backend Services project**: Manage a backend for app data and files storage, user management, cloud code, email and push notifications. This item is highlighted with a red rectangular border.
- Analytics project**: Discover usage patterns, analyze user data, log exceptions, trace problems and monitor app performance.
- AppFeedback project**: Receive and manage in-app feedback coming directly from your app.
- AppPrototyper project**: Build interactive high-fidelity prototypes for your mobile apps. Export them to AppBuilder. A small blue 'BETA' badge is visible.

# And just like that ..

# Your Backend Services Dashboard is ready

The screenshot shows the 'Configure: Services' section of a dashboard. On the left, there's a sidebar with a 'Services' section. The main area is titled 'Get started with Services' and lists several service categories, each with a brief description and an 'Add to project' button:

- Cloud Data**: Easily store, access, and manage the data for your app in the cloud. [Add to project](#)
- Push Notifications**: Send push notifications to iOS, Android, Windows Phone, and Windows devices. [Add to project](#)
- Responsive Images**: Optimize your images for a specific size or screen resolution and for faster download. [Add to project](#)
- Cloud Files**: Upload files to the cloud and serve them to users faster through a CDN for smoother user experience. [Add to project](#)
- User Management**: Add user authentication, authorization, and management to your app with Facebook, Google+, Active Directory etc. [Add to project](#)
- Data Connectors**: Use your existing data by exposing it directly to the app via our powerful cloud services. [Add to project](#)
- Cloud Code**: Execute custom business logic and validation rules straight on our servers. [Add to project](#)
- Email Notifications**: Write once, send to everyone. Use email templates when you want to send automated emails. [Add to project](#)

# See that API Key?

## It's important and unique to your Backend

The screenshot shows the 'Overview: API Keys' page in the Backend Services interface. The left sidebar has a 'API Keys' section selected under 'Overview'. The main content area displays three key components:

- Application Name:** A text input field containing 'DataConnector Demo'. Below it is a descriptive text: 'This is the name of your app in Backend Services. Your name is unique only in the scope of your own apps.'
- API Key:** A redacted text input field. Below it is a descriptive text: 'The API Key is the unique identifier of this application. Use it whenever you need to connect to this application.'
- API Master Key:** A text input field with a tooltip explaining its purpose: 'The Master Key can be used to bypass application security and grant you access to manage your Backend Services app from a service similar to the Backend Services portal. Don't deploy it with your app.' Below the tooltip is a 'View API Master Key' button.



## #1. Let's talk SDKs

You get wrappers .. over a consistent REST API

1. JavaScript SDK
2. .NET SDK
3. iOS SDK
4. Android SDK
5. Cloud Code

That's pretty much everything ..

Yup, that's the goal

Native SDKs make it easy to work with  
Backend Services REST API

Makes for smooth Integrations

Any App, Any Platform



## #2. Let's talk about Data

1. Can't be local in devices only
2. Users will want to roam between platforms/devices
3. Data types - Relational & Non-relational
4. Data needs exposed through clean APIs
5. Security & Scalability concerns

# Telerik Backend Services

Complete, Fast & Robust  
Mobile Backend as a Service

Your data however you want it

Scalable BaaS in the cloud

# Backend Content Types

Like a Table in a Relational Database

Only in the cloud & accessible

Content has **Fields** that describe the Data

Like column schema in a Table

# Backend Content Fields

TELERIK BACKEND SERVICES TYPE	DESCRIPTION
Id	Unique identifier of each record.
Text	String able to hold both text and objects. Note: Objects won't be displayed properly in the backend interface. Use an Object field if you need to store complex nested values.
Number	Number field.
Yes/No	Boolean field.
Date and Time	Date field.
File	Unique identifier of a file present in the Files type.
Geo Point	Plain JS object { "longitude": number, "latitude": number }
Object	An object field can be used to store complex nested values (ex. a nested object, array of objects etc.).
Array	Array used to hold multiple values or objects.
Relation	Pointer to the Id of an external type.
Relation - multiple	An array of pointers to the Ids of an external type.

Oh, and Content can be Related .. like Table relationships

# Creating Content

Create a content type

Type name  
Activities  
Instruction: Spaces are not allowed

Fields of this content type (you can define fields later)

Id	Identifier
CreatedAt	DateTime
ModifiedAt	DateTime
CreatedBy	Users
ModifiedBy	Users
Owner	Users

Display field (optional) (?)  
- None -

Add a field

Save Cancel

There are some default Fields .. you set up the rest

# Manage your Content

The screenshot shows the 'Data: Types' section of the Friends App Backend. The left sidebar has sections for Overview, Statistics, API Keys, Data (selected), Types (selected), Permissions, Data Connectors, Files (File Browser, Responsive Images, Permissions), and Push Notifications (Push Browser, Devices, Settings). The main area shows a table with two rows: 'Activities' (3 items, Backend Services (Cloud)) and 'Comments' (3 items, Backend Services (Cloud)). There are buttons for 'Create a Content Type' and 'Create Type from a Data Connector', and a 'Refresh' button.

Type	# ITEMS	STORAGE TYPE
Activities	3 items	Backend Services (Cloud)
Comments	3 items	Backend Services (Cloud)

All Content Types & Data - accessible through portal



## #3. Your Data, Our APIs

Best part about Backend Content = Easy APIs

REST API created auto-magically on top of Content

No need to write your own Services/APIs

REST API for any app, any platform

# Let's manage Data

Yes, you can do plain RESTful service calls

But SDKs make the integrations so much easier

Seamless CRUD operations!

# I see EverLive

Internal name for Backend Services

Used in namespaces & SDKs

Entry point into Backend Services SDK is through ..

The Everlive class

You normally create one global instance of 'Everlive'

Needs your Backend API Key

# Here's how you start

## .NET SDK

```
EverliveApp myApp = new EverliveApp("your-api-key-here");
```

## JavaScript SDK

```
var el = new Everlive('your-api-key-here');
```

# Read data from Backend

.NET code, granted you have a content called 'Activity'

## Single item by Id

```
public async Task GetSingleItem(EverliveApp app, Guid activityId)
{
    var activity = await app.WorkWith().Data<Activity>().GetById(activityId)
        .ExecuteAsync();
    Debug.WriteLine("Activity: " + activity.Text);
}
```

## Multiple items

```
public async Task GetMultipleItems(EverliveApp app)
{
    var activities = await app.WorkWith().Data<Activity>().GetAll().ExecuteAsync();
    foreach(var activity in activities)
    {
        Debug.WriteLine("Activity: " + activity.Text);
    }
}
```

# Manipulate data from Backend

.NET code, granted you have a content called 'Activity'

## Update

```
public async Task UpdateActivityById(EverliveApp app, Guid activityId,
                                      Activity updatedActivity)
{
    await app.WorkWith().Data<Activity>().Update(activityId, updatedActivity)
        .ExecuteAsync();
}
```

## Delete

```
public async Task DeleteById(EverliveApp app, Guid id)
{
    await app.WorkWith().Data<Activity>().Delete(id).ExecuteAsync();
}
```



## #4. Let's talk Data Connectors

We know you have existing data

Variety of DB providers = disparate content

Painful Service layers to expose data

It's time to bridge your On-premise data

You'll love this

# Data Connector flexibility

## Data Connectors

Microsoft SQL



Built-in MS SQL data connector allows you to seamlessly connect your mobile app to your existing MS SQL database.

MySQL



Easily connect your mobile app to your existing MySQL database using the built-in MySQL Data Connector.

Oracle



Connecting your mobile app to an existing Oracle database has never been easier with our built-in Oracle Data Connector.

PostgreSQL



PostgreSQL Data Connector enables you to easily connect your mobile app to an existing PostgreSQL database.

Set your on-premises data free ..

# How does it work?

You download & install an IIS app ..

It's the Data Link Server

Simply point to your on-premise Data storage

You create Mapped Content

Your data stays in house .. uses Backend API

Free REST API over your data!

# Don't code, just Configure!

Create Data Connector

Test connection succeeded.

What is a Data Connector?  
A Data Connector defines the connection to a single data store in a remote Data Link Server. [Read more...](#)

Data Connector Name  
NorthwindDataConnector

Example: *MyOldSQLDatabase*

Type  
Microsoft SQL Server

Data Link Server URL  
`http://57014f78.ngrok.com/`

Example: *http://datalinkserver.com:9090*

Authentication Key  
This key is used to secure the communication between your Telerik Backend Services project and the Data Link Server.  
[read more](#) [Download](#)

Connection String  
`Server=TelerikVM;Database=northwind;Integrated Security=True`

Example: *Server=myServerAddress;Database=myDB;User Id=myUsername;Password=myPassword;*

[Save](#) [Test](#) [Back](#) [Cancel](#)

Simply make your Data Link Server accessible  
And use Connection String to point to your data ..



## #5. Got Non-Relational data?

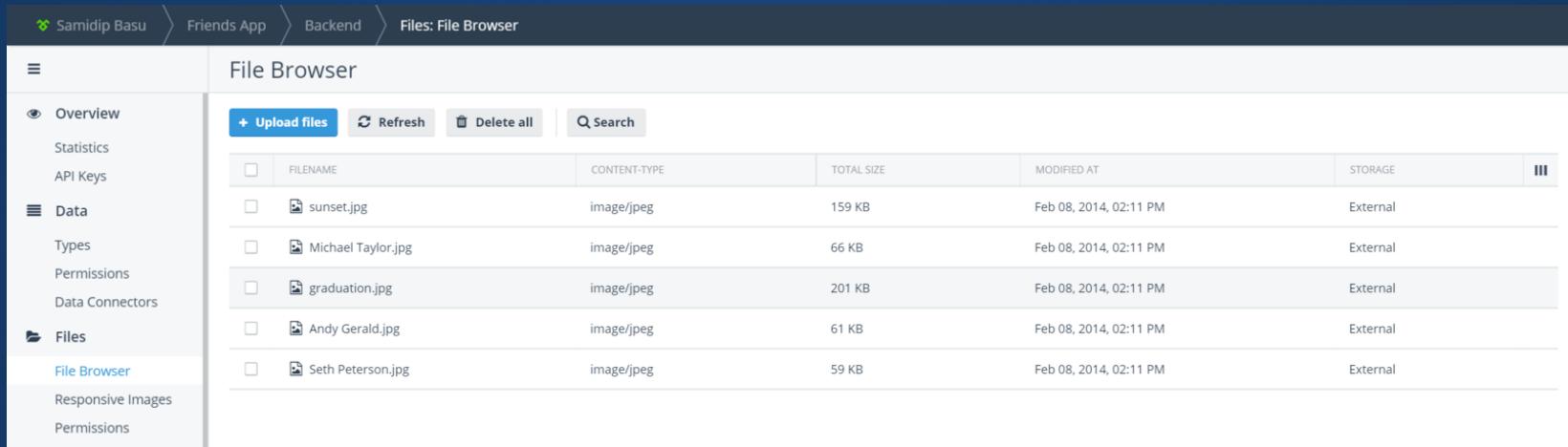
Yep, Backend Services allows you deal with **Files**

Think of it as Blob storage in the Cloud

And it's super scalable

Same RESTful API for your **Files**

# Backend portal helps manage Files



The screenshot shows a navigation bar at the top with 'Samidip Basu', 'Friends App', 'Backend', and 'Files: File Browser'. On the left, a sidebar has sections for Overview, Statistics, API Keys, Data (Types, Permissions, Data Connectors), and Files (File Browser, Responsive Images, Permissions). The main area is titled 'File Browser' and contains a table with columns: FILENAME, CONTENT-TYPE, TOTAL SIZE, MODIFIED AT, and STORAGE. The table lists five files: sunset.jpg, Michael Taylor.jpg, graduation.jpg, Andy Gerald.jpg, and Seth Peterson.jpg, all of which are image/jpeg type files stored externally.

FILENAME	CONTENT-TYPE	TOTAL SIZE	MODIFIED AT	STORAGE
sunset.jpg	image/jpeg	159 KB	Feb 08, 2014, 02:11 PM	External
Michael Taylor.jpg	image/jpeg	66 KB	Feb 08, 2014, 02:11 PM	External
graduation.jpg	image/jpeg	201 KB	Feb 08, 2014, 02:11 PM	External
Andy Gerald.jpg	image/jpeg	61 KB	Feb 08, 2014, 02:11 PM	External
Seth Peterson.jpg	image/jpeg	59 KB	Feb 08, 2014, 02:11 PM	External

Users can take pictures & upload to Backend  
2 lines of code for developers from any SDK

File delivery powered by global CDNs .. it's free!



## #6. Let's talk User Management

Traditionally, bothersome for developers ..

User authentication/authorization = Tricky

Account management = Security concerns

Backend Services to the rescue

# User Management - Elevated

1. New user registration
2. Automatic email flow for newly-registered accounts
3. Fine-grained control over User accounts
4. User Password management flows
5. User authentication through OAuth providers (Facebook, Google+, Live Connect, Twitter)
6. User authentication through Active Directory Federation Services (AD FS) authentication provider
7. Account deletion

# User Management in the Cloud

The screenshot shows a user management interface titled "Users Browser". The left sidebar contains navigation links for Overview, Statistics, API Keys, Data (Types, Permissions, Data Connectors), Files (File Browser, Responsive Images, Permissions), Push Notifications (Push Browser, Devices, Settings), and Users (Users Browser, Structure, Permissions, Roles, Automated Emails, Email Settings, Authentication). The "Users Browser" link is currently selected. The main content area displays a table of users with columns: USERNAME, EMAIL, ROLE, PROVIDER, VERIFIED, DISPLAY NAME, and MODIFIEDAT. The data in the table is as follows:

	USERNAME	EMAIL	ROLE	PROVIDER	VERIFIED	DISPLAY NAME	MODIFIEDAT
<input type="checkbox"/>	samidip	samidip@live.com	Registered	Everlive	No	Sam Basu	Jul 22, 2014, 09:20 AM
<input type="checkbox"/>	seth	seth.peterson@telerik.com	Registered	Everlive	No	Seth Peterson	Feb 08, 2014, 02:11 PM
<input type="checkbox"/>	michael	michael.taylor@telerik.com	Registered	Everlive	No	Michael Taylor	Feb 08, 2014, 02:11 PM
<input type="checkbox"/>	andy	andy.gerald@telerik.com	Registered	Everlive	No	Andy Gerald	Feb 08, 2014, 02:11 PM

Automated Email Templates .. everything on portal



## #7. Let's talk Push Notifications

Push Notifications are a great way to engage users

Present on every Mobile platform

Captures user attention even when app not running

It should be a no-brainer, right?

# Push Notifications are actually hard!

Configuring Push Notifications is tough for developers

Every Mobile platform does it differently

Apple, Google, Microsoft - all have own rules & setup

Backend Services is here to help

# Push Notifications in Backend Services

1. Works X-Platform for iOS, Android & Windows
2. Powerful targeting to filter on devices
3. Smart grouping on users/geography
4. Works for native or hybrid mobile apps

# How to work Push Notifications

1. Register device for Push from mobile app
2. Enable Push Notifications per platform
3. Build Segments for Filtering - user/device
4. Send Push Notifications from Backend
5. Push can also be triggered from REST endpoint

# Device Registration for Push

All this achieved with 2 lines of code

.NET

```
await app.WorkWith().Push().CurrentDevice.Initialize(PushSettings)
    .ExecuteAsync();
var result = await app.WorkWith().Push().CurrentDevice.
    Register(DeviceParameters customParameters)
    .ExecuteAsync();
```

JavaScript

```
var el = new Everlive('your-api-key-here');
el.push.register(
    pushSettings,
    function successCallback(data) {},
    function errorCallback(error) {}
);
```

# Device list on portal

The screenshot shows a user interface for managing devices. At the top, there is a navigation bar with the following items: Samidip Basu, Friends App, Backend, Push Notifications: ..., and a search bar. Below the navigation bar is a sidebar with the following sections and their sub-items:

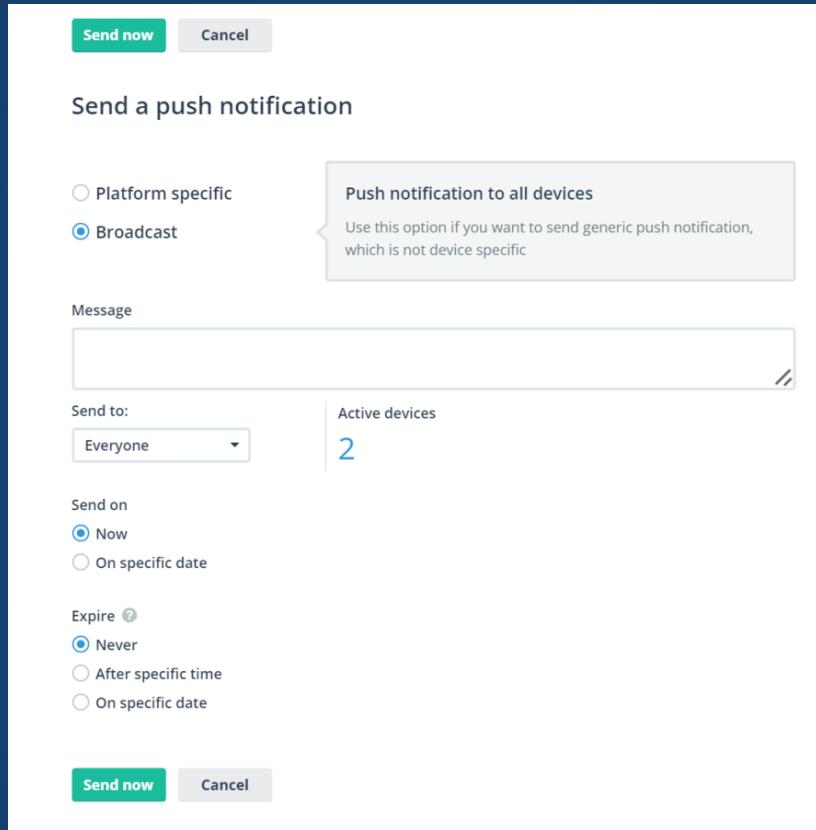
- Overview: Statistics, API Keys
- Data: Types, Permissions, Data Connectors
- Files: File Browser, Responsive Images, Permissions
- Push Notifications: Push Browser, Devices (which is the active section), Settings

The main content area is titled "Devices". It features a toolbar with Refresh, Delete all, and Search buttons. Below the toolbar is a table with the following columns: PLATFORM TYPE, PLATFORM VERSION, HARDWARE MODEL, and ACTIVE. There are two entries in the table:

PLATFORM TYPE	PLATFORM VERSION	HARDWARE MODEL	ACTIVE
WindowsPhone	8.0.9903.0	XDeviceEmulator	Yes
Windows	8	Parallels Software International Inc._Parallels Virtual Platform	Yes

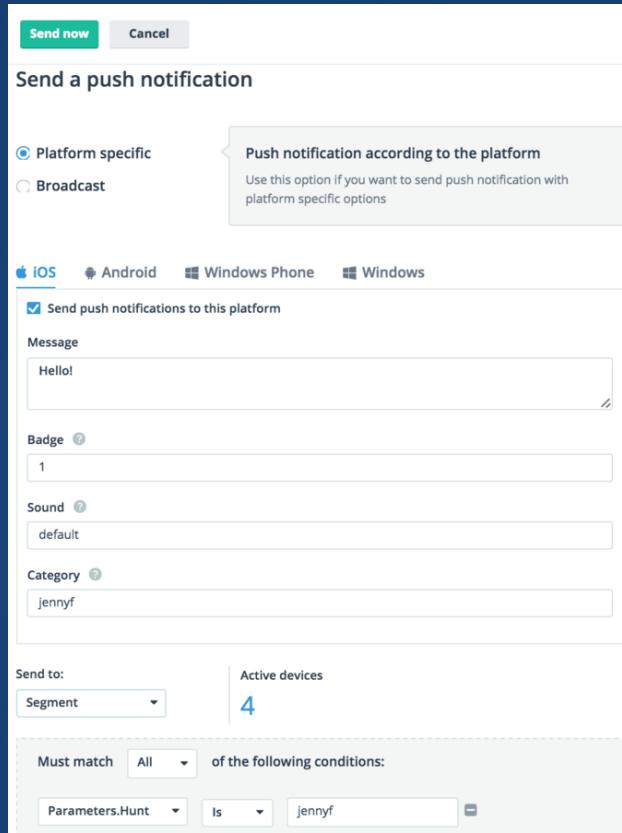
Manage registered devices .. even works on simulators

# Sending Push Notifications from portal



Broadcast to all devices or be Platform specific  
Payload can be tweaked for iOS/Android/Windows

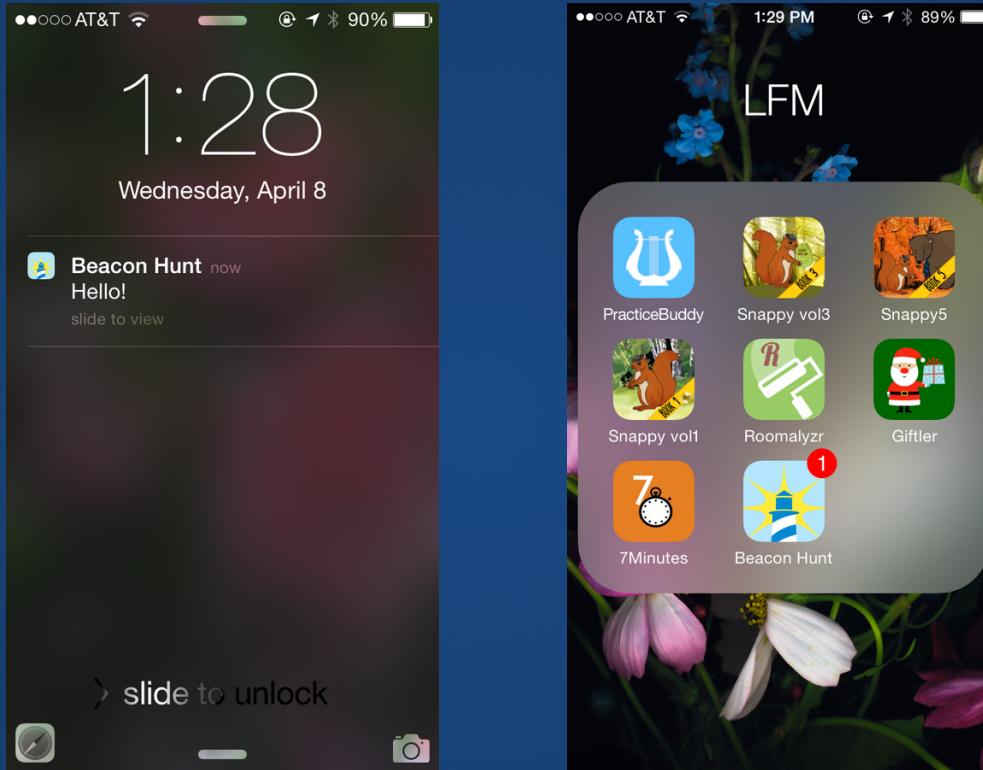
# Sending Push Notifications from portal



Here's an iOS specific Push Notification

Parameters in Segments allows you to filter devices

# Push Notifications in action!



Works on Devices or Simulators

Can be triggered through a REST endpoint



## #8. Let's talk Cloud Code

For times when you need custom logic server-side  
Simplicity of JavaScript  
Yes, this is based on Node.js

Built-in inside Backend Services

# Types of Cloud Code

Content Type Code - like DB triggers | works on content | acts before/after CRUD operations

Cloud Functions - custom JavaScript code | each function gets REST endpoint | invoke anytime

You can write Cloud Code ..

Directly in Backend or Upload custom JS file



## #9. Let's talk Responsive Images

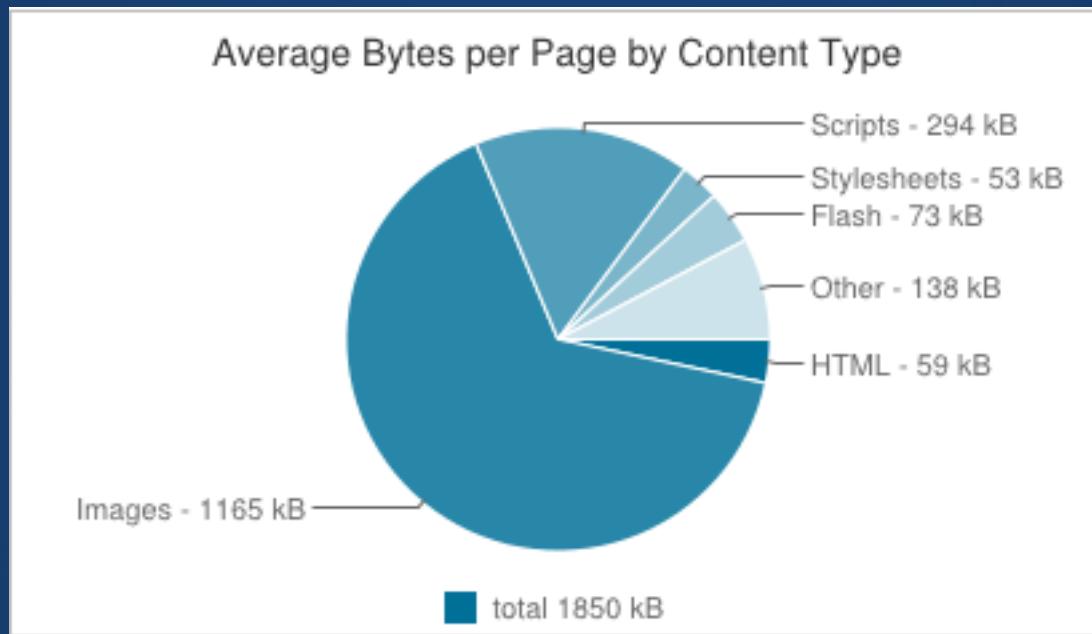
Images are selling points of many apps

Yet, ill-sized images are the biggest bandwidth hogs

Makes whole app feel sluggish

Backend Services does the heavy work

# Here's why Images are important



Typical HTTP bandwidth usage

Critical to optimize in Mobile first world

# Responsive Images in Backend

1. Automatic image re-sizing on server
2. Caters to exact device form factor
3. Bandwidth & app optimization
4. Support for high DPI retina devices
5. Platform agnostic
6. Image delivery over global CDNs
7. Simple JavaScript to set up

# Simply upload an Image to Backend



The screenshot shows a user interface for managing files. On the left, there's a sidebar with navigation links: Overview, Statistics, API Keys, Files (with sub-links: File Browser, Responsive Images, Permissions), Configure, and Services. The main area is titled "File Browser" and contains a "Refresh" button, a search bar, and a message stating "No items have been added yet". To the right, there's a modal window titled "Upload files" with a "Save" button, a "Cancel" button, and a "Maximize" button. The modal includes a dashed-dotted drop zone with the text "Drag and drop files here or Browse for files".

Works even if you are self-hosting images ..

```
https://bs1.cdn.telerik.com/image/v1/your-api-key-here/  
    resize=rezise-options-here/URI-to-image-here
```

URI to Responsive Image .. like magic!

That's a lot of features ...  
Everything you need from a  
Modern, Robust, Scalable Backend

Feeling empowered?

Let's Recap

# Telerik Backend Services

1. Relational or Non-relational data in the cloud
2. Automatic REST API on top of data
3. Data Connectors for On-premise data
4. Wrapper SDKs for any app on any platform
5. Robust User Management
6. Easy X-Platform Push Notifications
7. Cloud Code for server-side logic & much more

Complete BaaS offering

# Telerik Backend Services

Get started today

<http://www.telerik.com/backend-services>

Go build your client app .. you Backend awaits!

 **Telerik** Backend Services