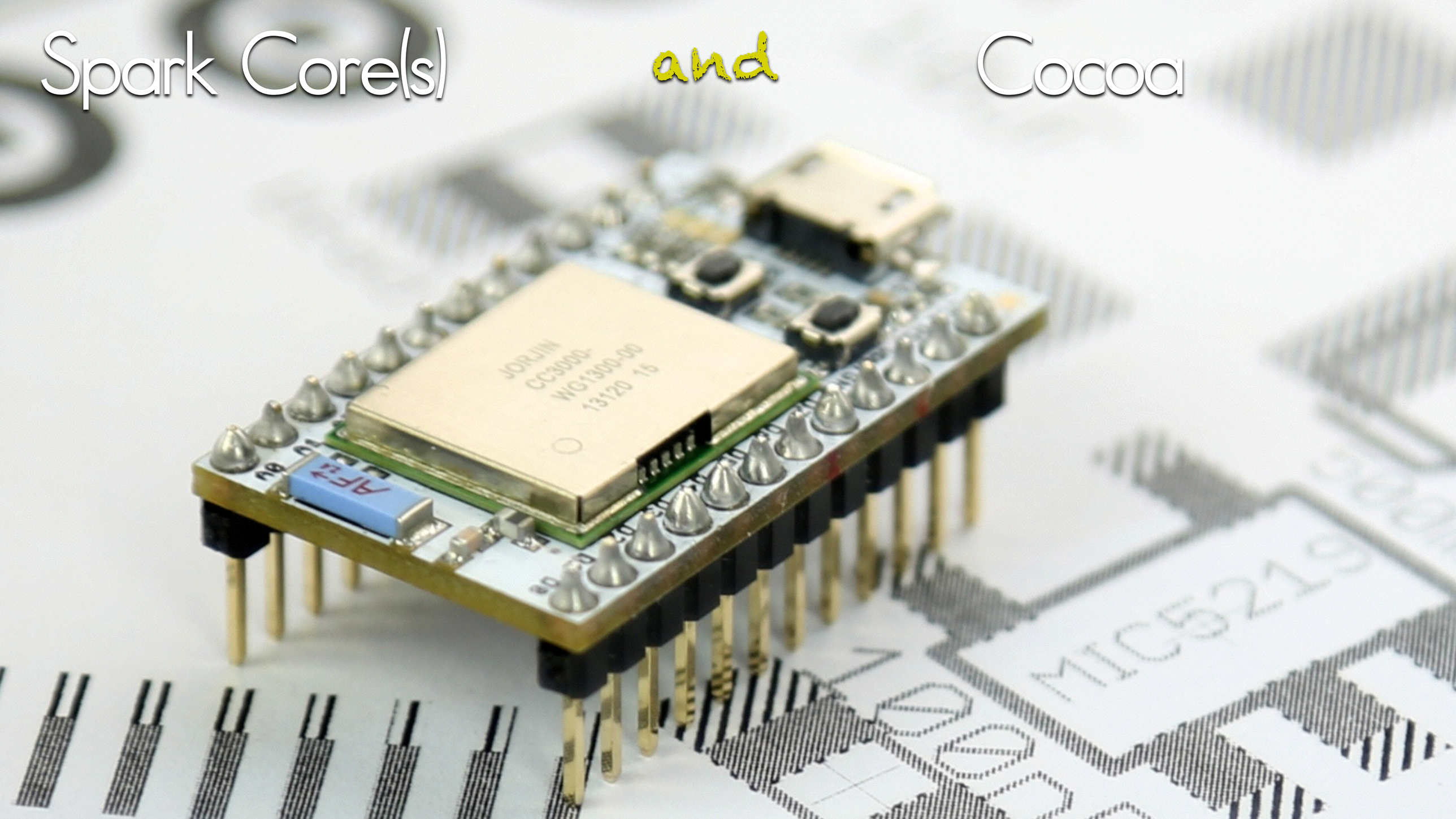
Spark Core(s) and Cocoa



Spark Core(s) and Cocoa

A spark Core is postage stamp-sized hackable Wi-Fi module for interacting with physical things.

- Easy accessable via Wi-Fi
- Arduino-based development environment
- Cloud-based API

A Spark Core can do almost all the things an Arduino can do, and has Cloud access.

Spark Core(s) and Cocoa

Handling

- Every core has a unique ID
- Every user has a unique ID, an email-dress.
- In addition, a user must have an access token, provided by the cloud
- A core needs to be claimed by a user

To access a Spark Core, you need the physical core with Wi-Fi access, an account and an access-token.

Spark Core(s) and Cocoa

Cloud access:

```
curl https://api.spark.io/v1/devices/
0123456789abcdef01234567/brew \
-d access_token=9876987698769876987698769876987698769876
```

Typical statement with device-ID and access-token.

Spark Core(s) and Cocoa

Coding the Core

Cloud API

2 different kind of calls, function calls and variables

Function calls are limited to 4 different functions with one argument.
The argument is a character string (not bytes!) with its length limited to 63 chars.
The return value of a function call is a single integer.

Names need not to identical, though it manes sense.

```
int brew(String args)
{
  // parse args
  // ...
  int status_code = ...
  return status_code;
}
```

```
void setup()
{
  Spark.function("brew", brew);
}
```

Spark Core(s) and Cocoa

Coding the Core

Cloud API

Variables

The name of variables is limited to 12 chars.

Type and name are exposed on the core to the cloud.

```
int temperature = 0;

{
...
  Spark.variable("temperature", &temperature, INT);
...
}
```

Spark Core(s) and Cocoa

The Core can more, like TCP and UDP.

IP adress and port name and some function calls
to open and close the port.

Direct contact without the cloud.
Basic TCP transfers, just a bunch of bytes.
UDP maybe unsafe.

Spark Core(s) and Cocoa

Coding with Cocoa

Two packages:
Spark Core iOS Library
(https://github.com/hoffmanjon/SparkCoreIOS
This packages handles only the cloud API

UDP
CocoaAsyncSocket
(https://github.com/robbiehanson/CocoaAsyncSocket)

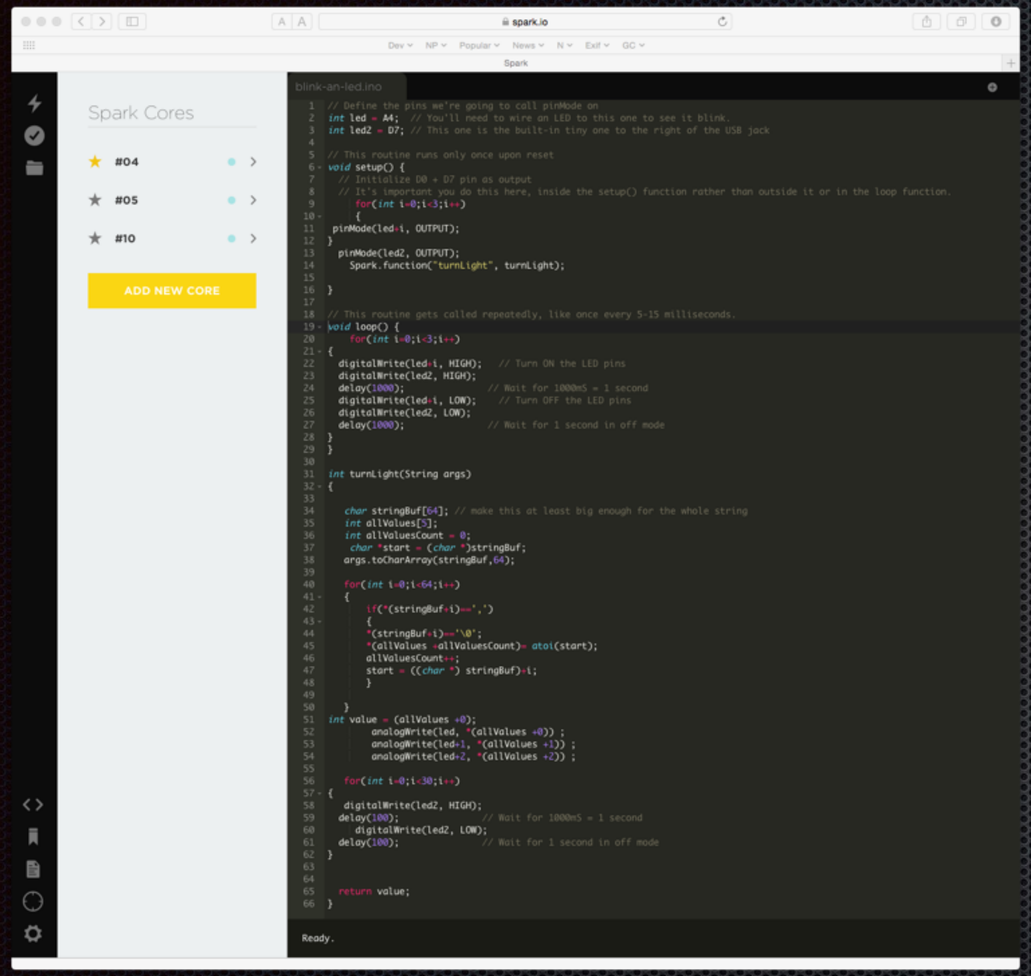
This package handles the UDP contact

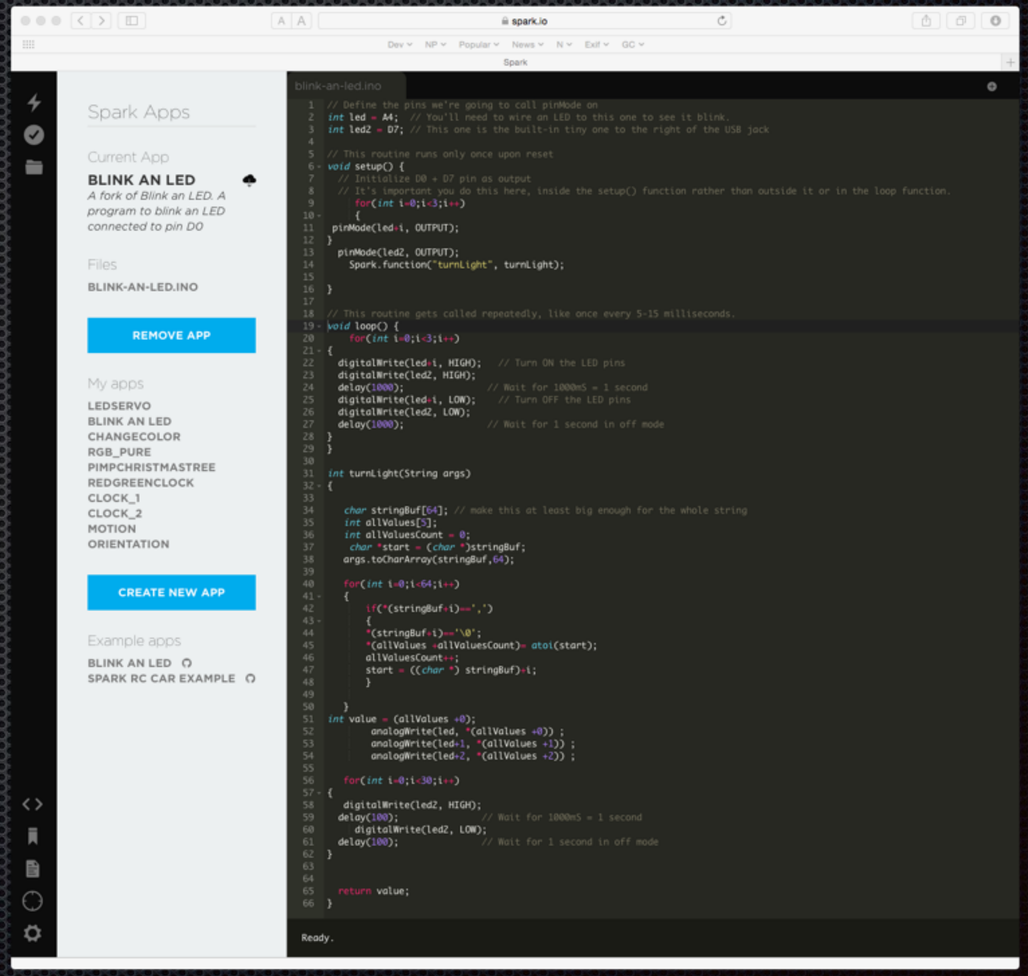
FTP is written based on Apples example code.

Programming

The only way to program a core is through the cloud API.
Code is written as Arduino-Language in the Web-interface.
The code is stored and compiled on the server.
In return the cloud API flashes the core, known by ID and token, with the compiled code.

Spark Core(s) and Cocoa





Spark Core(s) and Cocoa

Workflow

Launch the cores with the IP-Adress as variable and
an exposed function, then start the app.

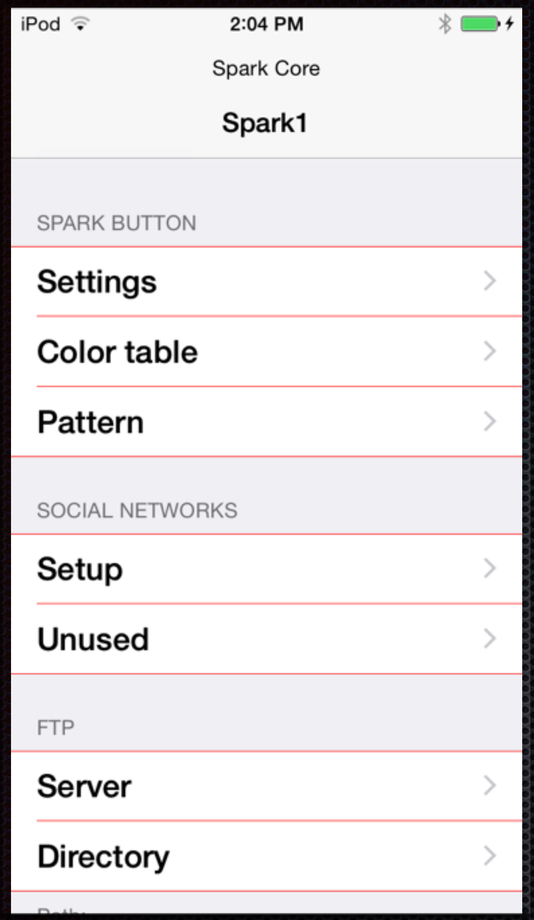
- Request the IP address from the core. (via cloud)

- Set the port number with a function call (via cloud)

- Establish the UDP contact.

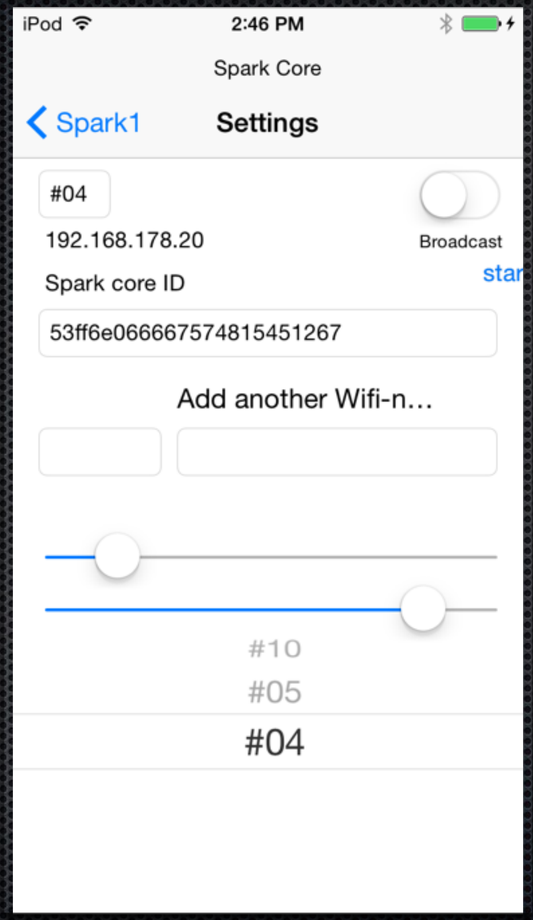
Here there is a general listening port for all the buttons, and
a dedicated port to send the color values

Spark Core(s) and Cocoa

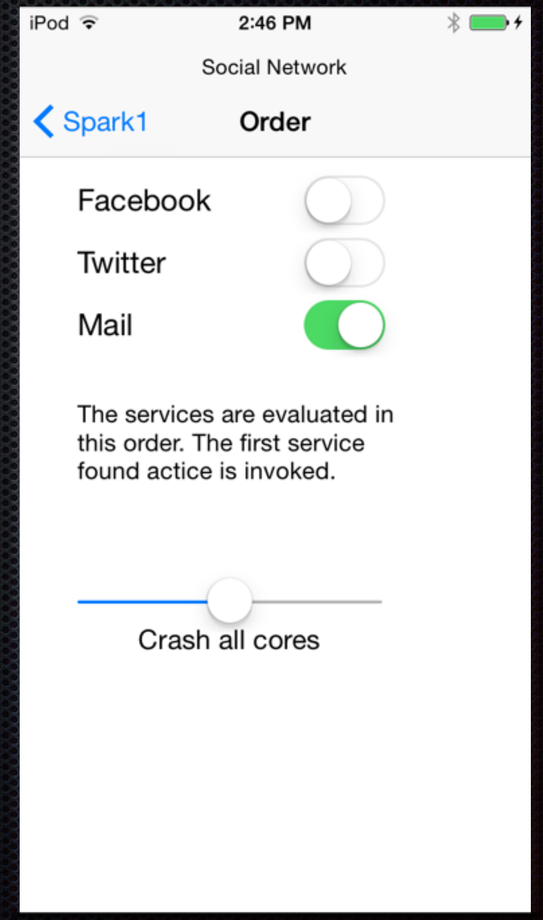


Main Screen

General



Spark Cores



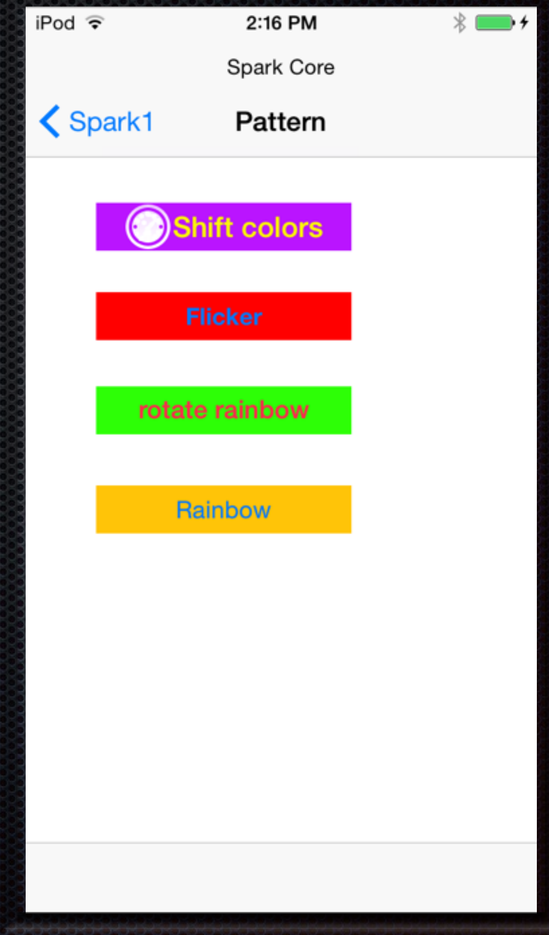
Prefs for Social

Spark Core(s) and Cocoa

All LED colors

Colors

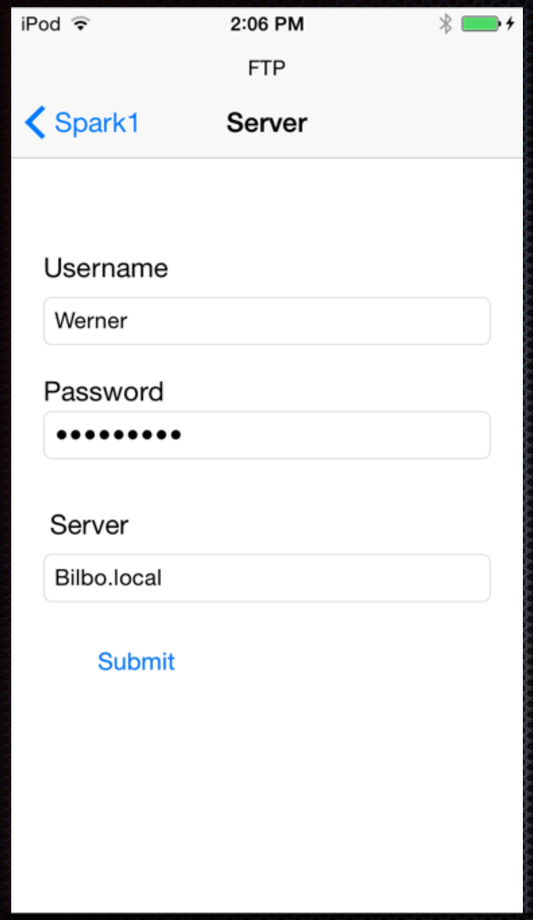
single Color



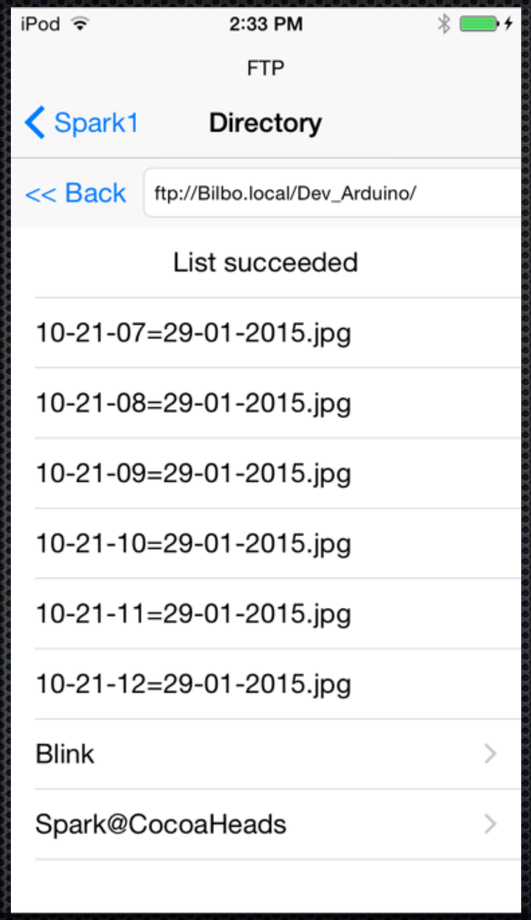
Pattern

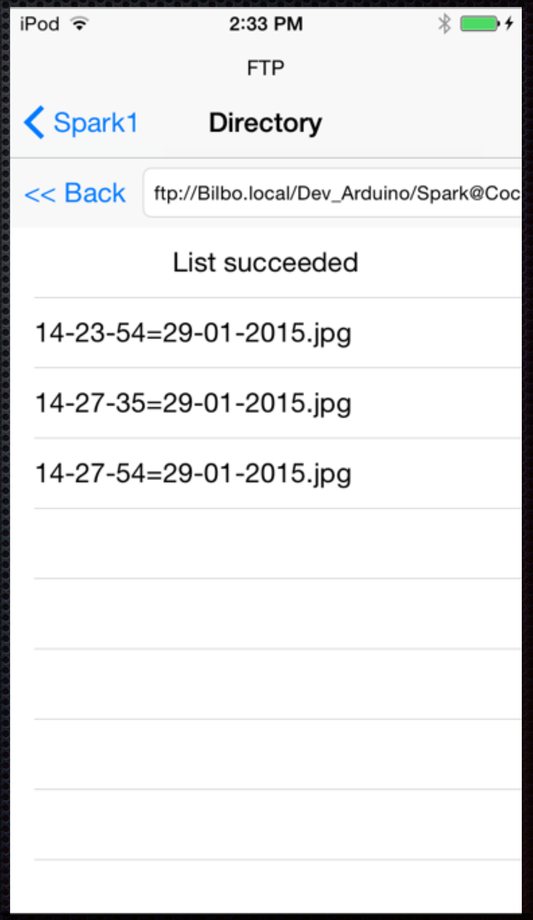
Spark Core(s) and Cocoa

FTP upload



FTP Server



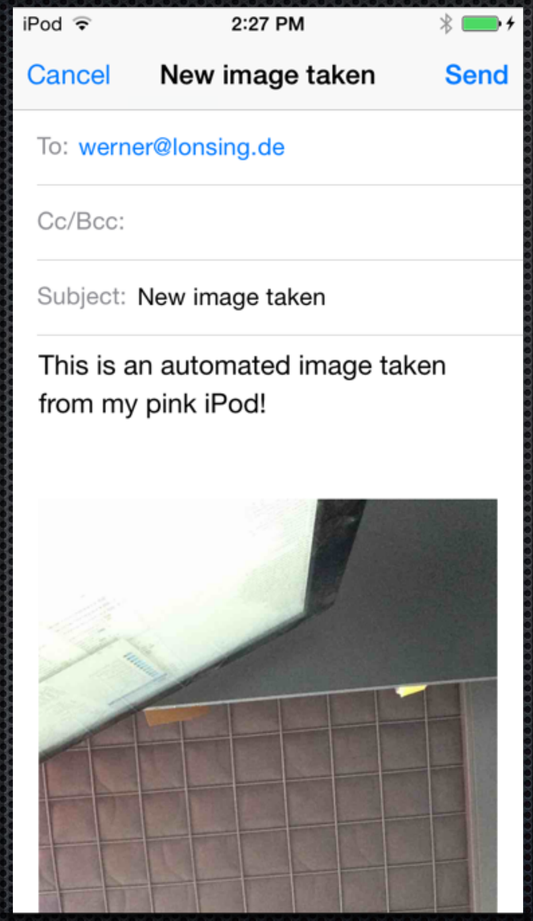


FTP Directories

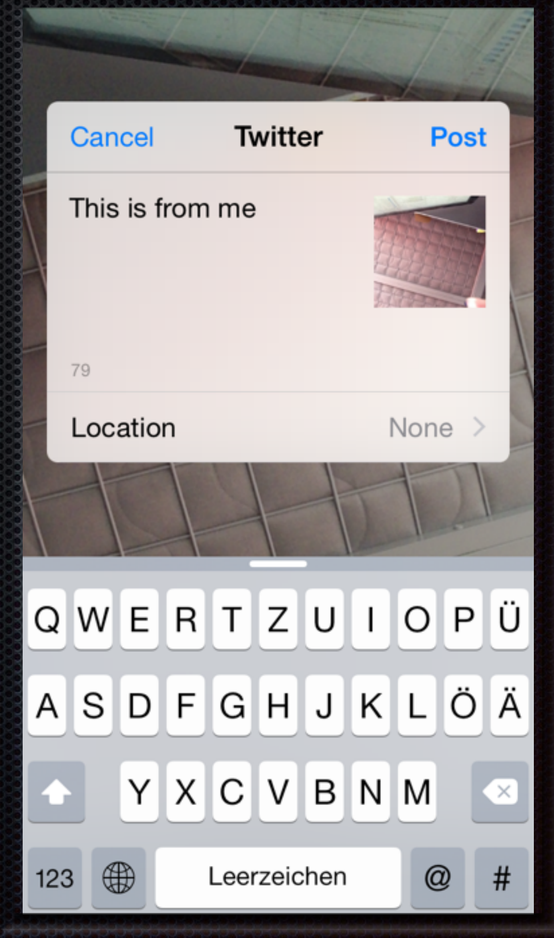
Spark Core(s) and Cocoa

Social

Facebook



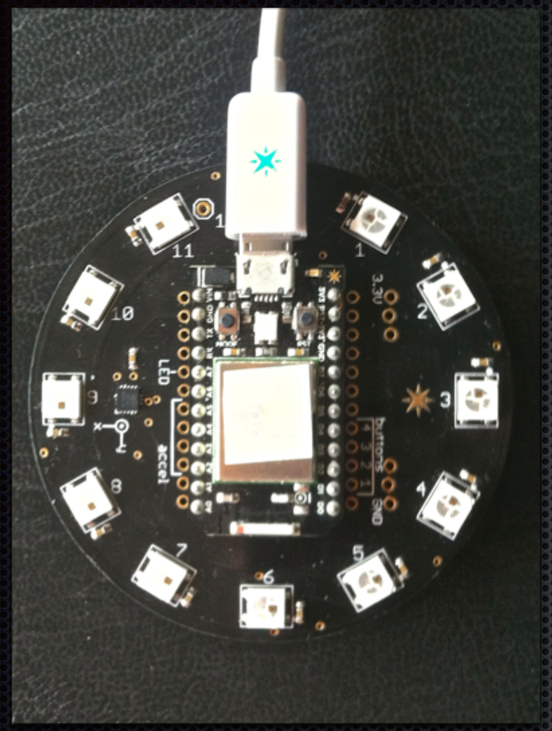
MailComposer



Twitter

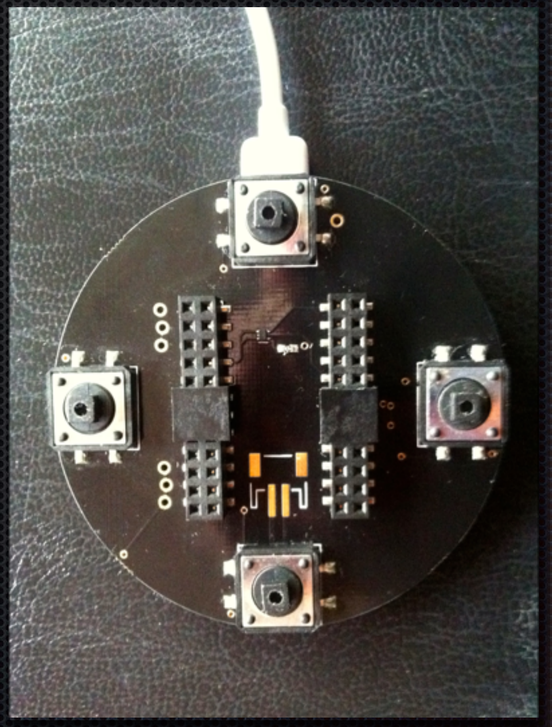
Spark Core(s) and Cocoa

Spark Button



The Spark core has 2 little buttons and a tiny RGB-LED.

The plate has 11 individually controllable RGB-LEDS, an 3-axis accelerometer and 4 tactile buttons for interactions



Front with 12 LEDs

Back with 4 Buttons

Interface of the SparkButton

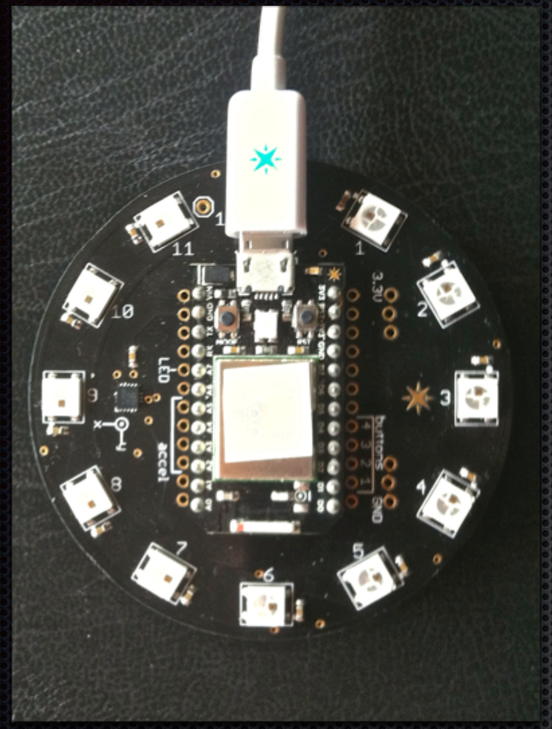
Spark Core(s) and Cocoa

Some demo, maybe

Spark Core(s) and Cocoa

Usage

mode
[remove picker]



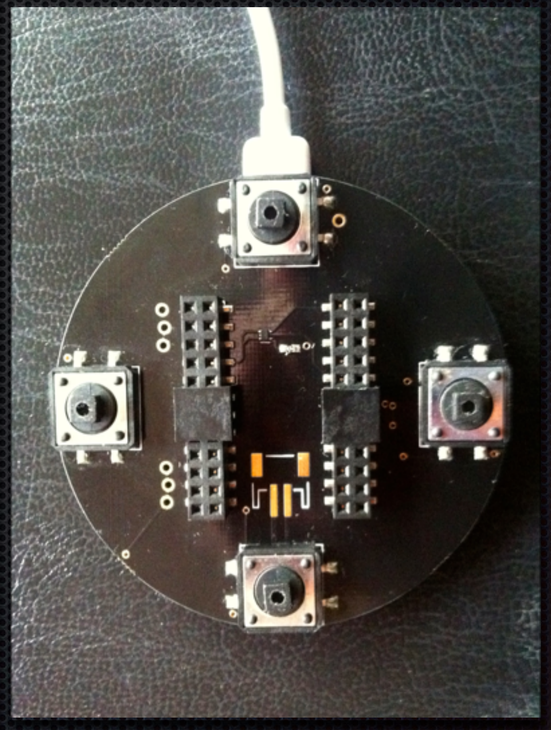
increase

decrease

The mode button
switches between brightness
and rotation on the core

[show picker]
take photo

decrease



increase

mode
[remove picker]

[show picker]
take photo

Usage of the 4 buttons

Spark Core(s) and Cocoa

Thank you!