apps

Find the latest version of this document at http://touchdevelop.com/slides .

learning objectives

- **UI:** interacting with users
- **web requests:** interacting with the web
- **libraries:** reuse code from others
- **data:** storing and reading structure data

UI

boxcraft

boxes

- nested vertical and horizontal boxes
- box attributes: color, border, width, height, margin, padding

```
boxed
  boxed
    box->use horizontal layout
    boxed
    boxed
    boxed
  boxed
```

pages

- the app has a stack of pages, only the topmost is visible
- **page has 2 parts**
  - **initialize**: code run once to setup data
  - **display**: code run every time something changes – build the boxes!

editing text

- “show a text box”

```
box->edit text(data->text, true)
```

- “do something when text changes”

```
box->on text editing(handler)
  where handler(text : string) is
      data->text := text
```

basic skinning

- set background and foreground colors

```
box->set foreground(colors->white)
box->set background(colors->black)
```

- give me some space

1 ‘em’ = height of ‘M’ in the current font

```
box->set margins(1,1,1,1)
box->set padding(1,1,1,1)
```

more skinning

- stretch horizontally

1 is a “weight”

```
box->set horizontal alignment(1, 1)
```

- arrange child boxes horizontally

```
box->use horizontal layout
```

buttons

- “do something when tapped”

```
box->on tapped(handler)
  where handler() is
    do something!
```

golden rules of boxcraft

"Thou shalt only read data when building boxes; thou shalt do anything in callbacks."

- yes!

  boxed

  ```
  data->text->post to wall
  ```

  reading data is allowed

- Yikes!

  boxed

  ```
  data->text := "can't write data"
  ```

  writing data is not allowed

web requests

web apis

download and parse

- download data

```
var text := web->download(url)
```

link

- parse data

```
var feed := web->feed(text) // rss, atom
var xml := web->xml(text)
var json := web->json(text)
```

more control

- fully featured HTTP requests and responses

```
var request := web->create request(url)
var response := request->send
```

- async for responsive UI

```
var request := web->create request(url)
request->on response received(handler)
    where handler(response)
        …
request->send async
```

geo-location

retrieve the longitude and latitude

- `var loc := senses->current location`
  - might use WiFi, cell tower, IP address, GPS
  - might have low precision 1 mile

libraries

reusing code

libraries

- a **library** is a script that can be ‘used’ by other script
  - **twitter search**
  - data.gov reader
  - text validators
  - …

demo

Let’s use the ‘twitter search’ library…

- open script editor,
- scroll down to libraries
- type `/mdvf`
- select ‘twitter search’

data

records

records

- structured data automatically saved
  - **table**: similar to your Excel table.
  - **index**: table with index lookup
  - **object**: garbage collected objects
  - **decorator**: add fields to existing objects

records

- programming task: store tweets in records