



Notes Apps auf iPad & Co

Wie native Apps mit Notes synchronisieren können.

Ich bin

- Dipl. Inf. Julian Buß
- Gründer von YouAtNotes in 2001, seitdem Entwicklung mit IBM Notes und Domino
- Seit 2013 freiberuflicher Entwickler und Berater (mehr unter julianbuss.net)
- Seit 2008 Entwicklung mit XPages
- Seit 2011 Entwicklung von nativen mobilen Apps
- Entwickler von Domino To Go, einem Framework in dem viele hier vorgestellte Techniken fertig nutzbar sind

Notes & Mobile

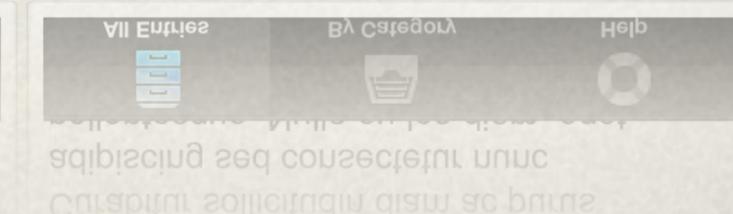
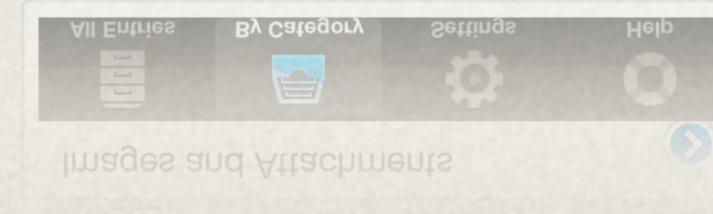
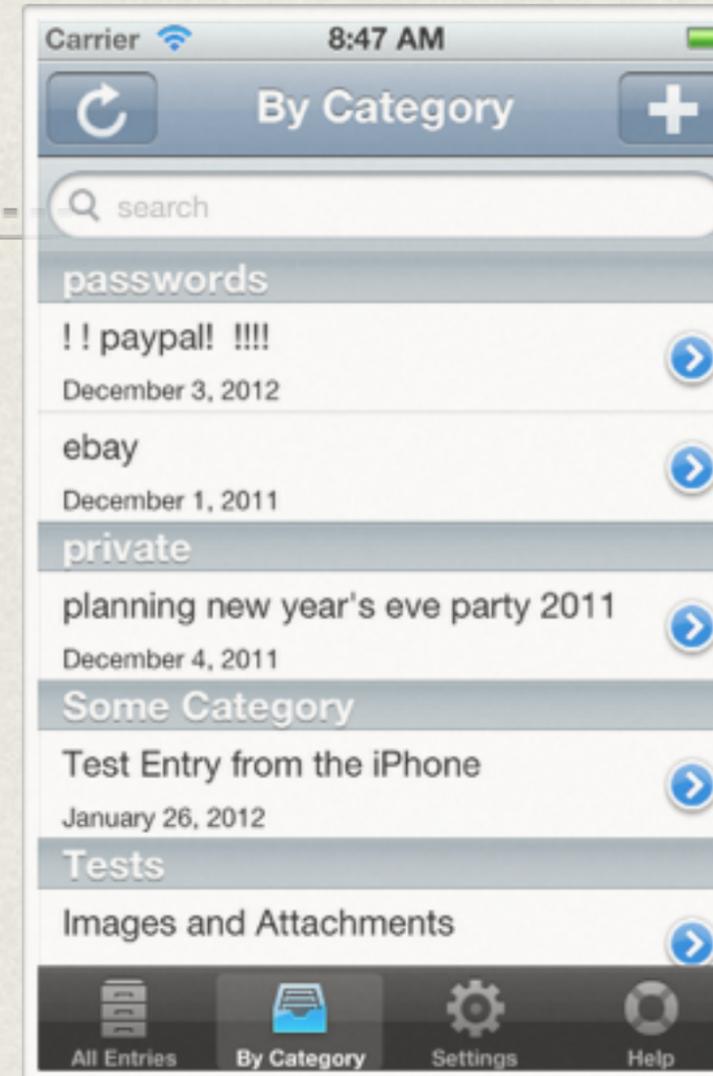
- beschäftigt mich seit über drei Jahren in Form von
- mobile Web Apps

Notes & Mobile

- beschäftigt mich seit über drei Jahren in Form von
 - mobile Web Apps
 - native Apps

Notes & Mobile

- beschäftigt mich seit über drei Jahren in Form von
 - mobile Web Apps
 - native Apps



Und so kann
das in echt
aussehen



Möglichkeit 1: Web-Apps

- + mit XPages realisierbar
- + Daten bleiben in Domino
- + kein Roll-Out, nur URL öffnen
- + mit vielen Geräten nutzbar

Möglichkeit 1: Web-Apps

- + mit XPages realisierbar
- + Daten bleiben in Domino
- + kein Roll-Out, nur URL öffnen
- + mit vielen Geräten nutzbar
- Gerät braucht immer Netz
- Login bei jedem Öffnen nötig
- langsames Netz = langsame App = traurige Benutzer
- fühlt sich häufig fremd an
- nativ aussehende Layouts für jedes Gerät aufwendig
- Zugriff auf Sensoren/Hardware schwierig (Fotos, GPS, Bewegung...)
- Speichern auf dem Gerät sehr eingeschränkt

Möglichkeit 1: Web-Apps

- + mit XPages realisierbar
- + Daten bleiben in Domino
- + kein Roll-Out, nur URL öffnen
- + mit vielen Geräten nutzbar

Hauptproblem: nicht offline fähig

- Gerät braucht immer Netz
- Login bei jedem Öffnen nötig
- langsames Netz = langsame App = traurige Benutzer
- fühlt sich häufig fremd an
- nativ aussehende Layouts für jedes Gerät aufwendig
- Zugriff auf Sensoren/Hardware schwierig (Fotos, GPS, Bewegung...)
- Speichern auf dem Gerät sehr eingeschränkt

Möglichkeit 2: native Apps

- + funktionieren immer
- + voll offline fähig
- + voller Zugriff auf Hardware und Sensoren (Fotos!)
- + Login kann in App gespeichert werden
- + nativer Look & Feel auf jedem Gerät
- + Antwortzeiten unabhängig vom Netz
- + fast beliebig viele Daten auf Gerät speicherbar

Möglichkeit 2: native Apps

- + funktionieren immer
- + voll offline fähig
- + voller Zugriff auf Hardware und Sensoren (Fotos!)
- + Login kann in App gespeichert werden
- + nativer Look & Feel auf jedem Gerät
- + Antwortzeiten unabhängig vom Netz
- + fast beliebig viele Daten auf Gerät speicherbar
- schwierig zu realisieren, fremde Entwicklungsumgebung, fremde Sprachen
- verschiedene Geräte = verschiedene Sprachen, Code teilen daher schwierig
- Verteilung erfordert mehr Aufwand

Möglichkeit 2: native Apps

- + funktionieren immer
 - + voll offline fähig
 - + voller Zugriff auf Hardware und Sensoren (Fotos!)
 - + Login kann in App gespeichert werden
 - + nativer Look & Feel auf jedem Gerät
 - + Antwortzeiten unabhängig vom Netz
 - + fast beliebig viele Daten auf Gerät speicherbar
- schwierig zu realisieren, fremde Entwicklungsumgebung, fremde Sprachen
 - verschiedene Geräte = verschiedene Sprachen, Code teilen daher schwierig
 - Verteilung erfordert mehr Aufwand
- Hauptproblem: fremde Sprache!*

Möglichkeit 2: native Apps

```
#import <Foundation/Foundation.h>

@interface Box:NSObject
{
    double length;    // Length of a box
    double breadth;  // Breadth of a box
    double height;   // Height of a box
}
@property(n nonatomic, readwrite) double height; // Property

-(double) volume;

@end

@implementation Box

@synthesize height;

-(id)init
{
    self = [super init];
    length = 1.0;
    breadth = 1.0;
    return self;
}

-(double) volume
{
    return length*breadth*height;
}

@end
```

- schwierig zu realisieren, fremde Entwicklungsumgebung, fremde Sprachen
- verschiedene Geräte = verschiedene Sprachen, Code teilen daher schwierig
- Verteilung erfordert mehr Aufwand

Hauptproblem: fremde Sprache!

Möglichkeit 2: native Apps

```
#import <Foundation/Foundation.h>

@interface Box:NSObject
{
    double length;    // Length of a box
    double breadth;  // Breadth of a box
    double height;   // Height of a box
}
@property(n nonatomic, readwrite) double height; // Property

-(double) volume;

@end

@implementation Box
@synthesize height;

-(id)init
{
    self = [super init];
    length = 1.0;
    breadth = 1.0;
    return self;
}

-(double) volume
{
    return length*breadth*height;
}

@end
```

```
public class DisplayMessageActivity extends Activity {

    @SuppressWarnings("NewApi")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_display_message);

        // Make sure we're running on Honeycomb or higher to use
        ActionBar APIs
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.HONEYCOMB)
        {
            // Show the Up button in the action bar.
            getActionBar().setDisplayHomeAsUpEnabled(true);
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
            switch (item.getItemId()) {
                case android.R.id.home:
                    // NavUtils.navigateUpFromSameTask(this);
                    return true;
            }
            return super.onOptionsItemSelected(item);
        }
    }
}
```

Hauptproblem: fremde Sprache!

Warum offline wichtig ist

Es gibt so viele Situationen...

Warum offline wichtig ist

Es gibt so viele Situationen...

- Unterwegs in Bahn, Auto, Flugzeug
- im Ausland (Roaming!)
- in besonderen Bereichen (Krankenhaus...)
- im Serverraum, der zufällig im Keller ist
- und so weiter...

Warum offline wichtig ist

Es gibt so viele Situationen...

- Unterwegs in Bahn, Auto, Flugzeug
- im Ausland (Roaming!)
- in besonderen Bereichen (Krankenhaus...)
- im Serverraum, der zufällig im Keller ist
- und so weiter...

Und genau dann will
man doch auf dem
Tablet oder Smartphone
arbeiten!

STREIFEN |

Warum offline wichtig ist

Es gibt so viele Situationen...

- Unterwegs in Bahn, Auto, Flugzeug
- im Ausland (Roaming!)
- in besonderen Bereichen (Krankenhaus...)
- im Serverraum, der zufällig im Keller ist
- und so weiter...

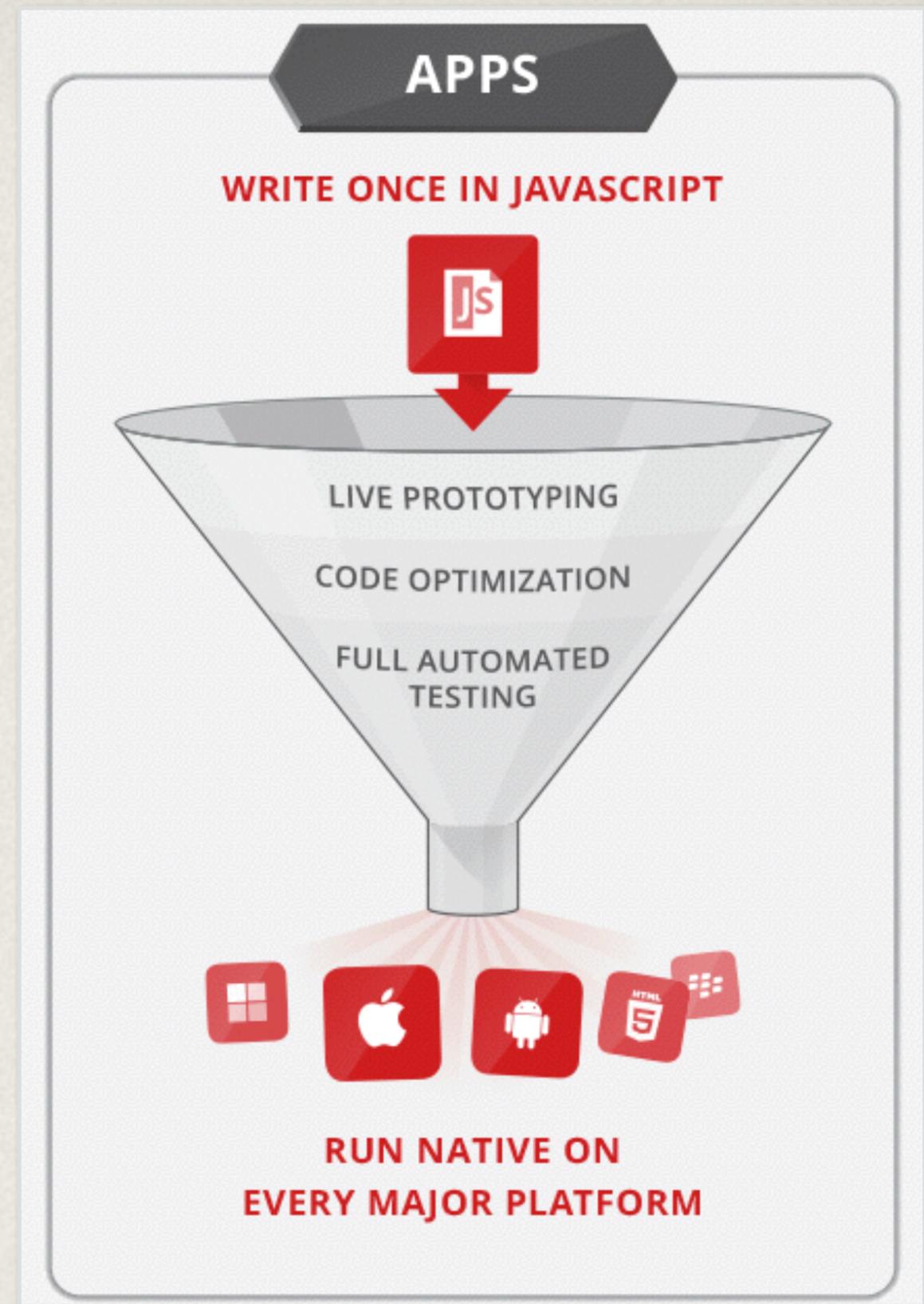
Also nützt eine Web-App dann genau: nix

Und genau dann will
man doch auf dem
Tablet oder Smartphone
arbeiten!

STREIFEN |

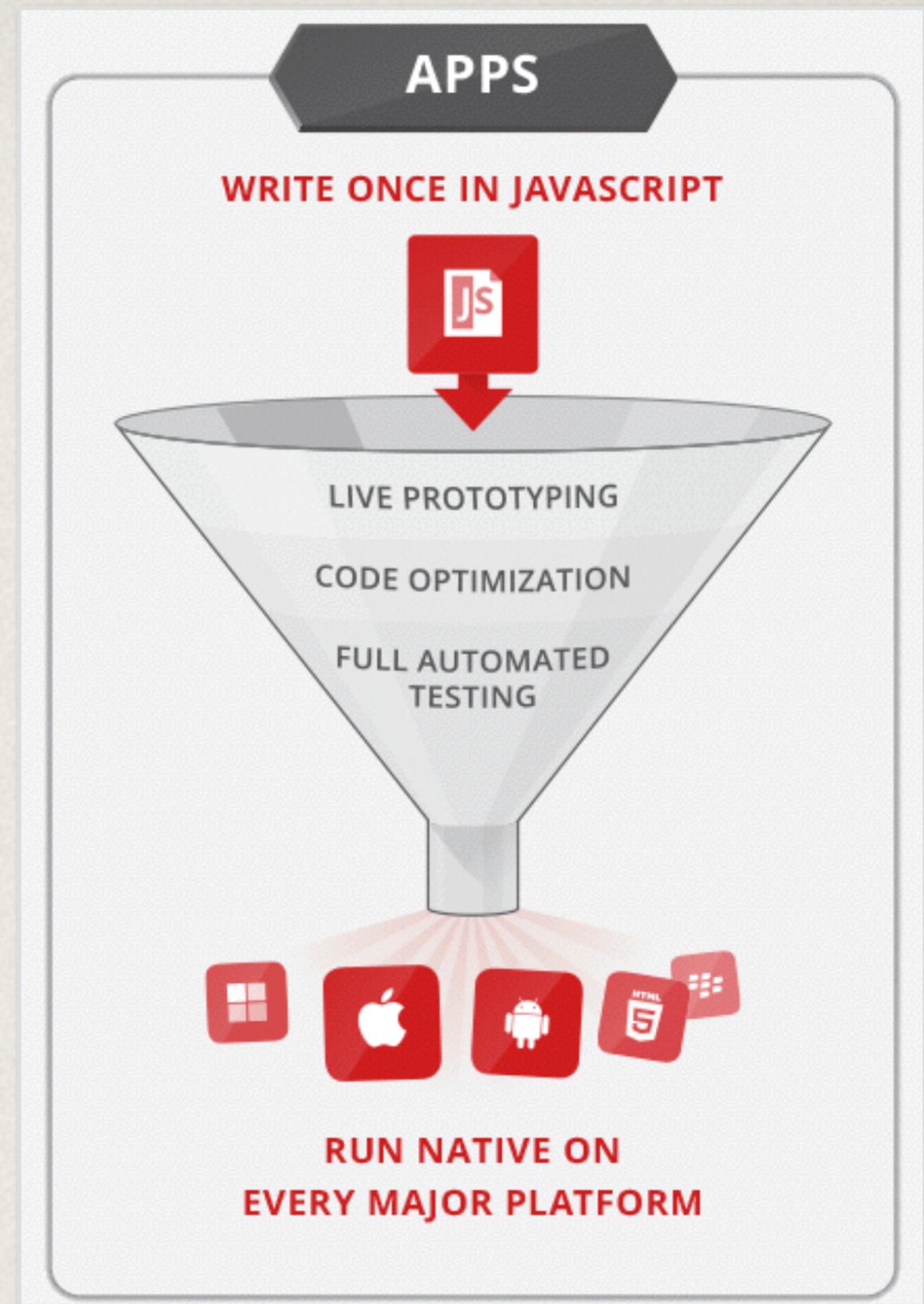
native App: aber wie?

- universelle Entwicklungsumgebung nutzen
- die Sprache nutzen, die man von XPages kennt: JavaScript
- unterschiedliche Plattformen ausgleichen mit einem Framework



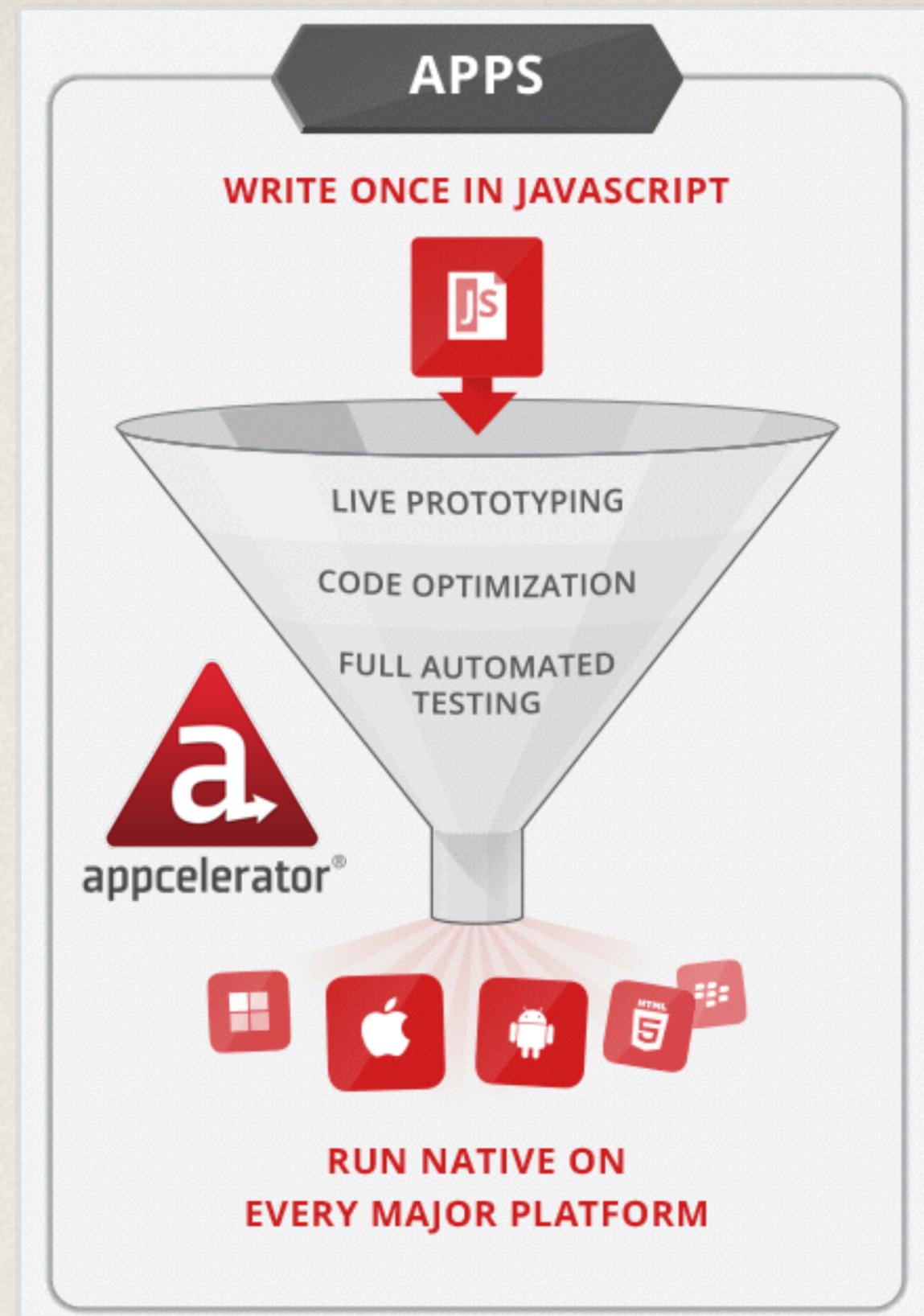
native App: aber wie?

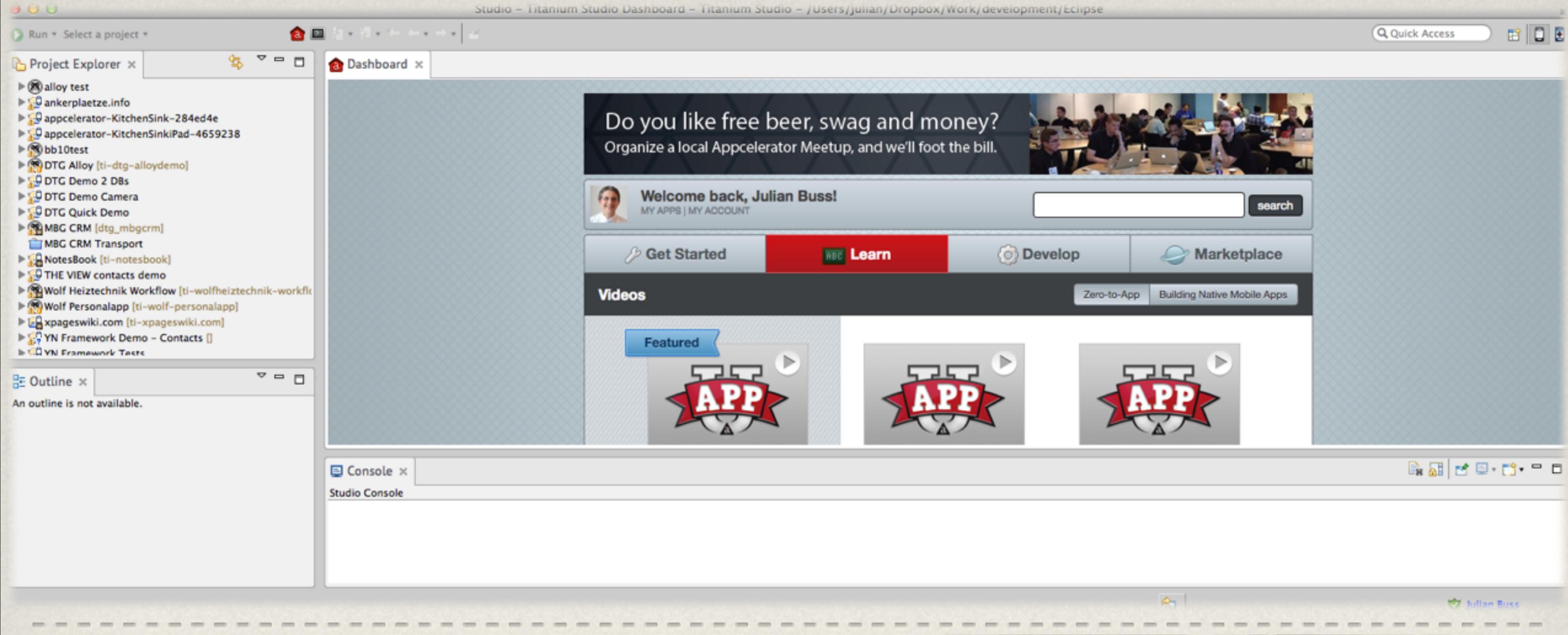
- universelle Entwicklungsumgebung nutzen
- die Sprache nutzen, die man von XPages kennt: JavaScript
- unterschiedliche Plattformen ausgleichen mit einem Framework
- meine Wahl: Appcelerator Titanium



native App: aber wie?

- universelle Entwicklungsumgebung nutzen
- die Sprache nutzen, die man von XPages kennt: JavaScript
- unterschiedliche Plattformen ausgleichen mit einem Framework
- meine Wahl: Appcelerator Titanium





Ein kurzer Blick auf Appcelerator Titanium

Nun aber zu Notes und Domino und XPages

Daten aus Notes und Domino...

- müssen vom Gerät **gelesen** werden
- und auf dem Gerät **gespeichert** werden
- und auf dem Gerät **verarbeitet** werden
- und Änderungen vom Gerät wieder **zu Domino hochgeladen** werden

Daten aus Notes und Domino...

- müssen vom Gerät **gelesen** werden
- und auf dem Gerät **gespeichert** werden
- und auf dem Gerät **verarbeitet** werden
- und Änderungen vom Gerät wieder zu Domino **hochgeladen** werden

Aber wie? Und auch noch offline fähig?

Daten aus Notes und Domino...

- müssen vom Gerät **gelesen** werden
- und auf dem Gerät **gespeichert** werden
- und auf dem Gerät **verarbeitet** werden
- und Änderungen vom Gerät wieder zu Domino **hochgeladen** werden

Aber wie? Und auch noch offline fähig?

Genau das klären wir hier!

Datenaustausch - Grundlagen

- Netz? -> Transport via Internet, Domino in DMZ oder intern
(Sicherheit: Authentifizierung, Verschlüsselung, VPN)
- Protokoll? -> HTTP bzw. HTTPS
(bewährt, unterstützt von Domino/XPages und Titanium)
- Format? -> JSON und ggf. XML
(JSON=Standardformat in JavaScript Welt, unterstützt von Domino/XPages und Titanium / XML ebenfalls auf beiden Seiten gut unterstützt)

Schritt 1: Daten lesen

Domino Format

```
{
"@timestamp": "20140227T165433,89Z",
"@toplevelentries": "96",
"viewentry": [
{
"@position": "1",
"@unid": "C125796E00337519802572570066BA1F",
"@noteid": "9EE",
"@siblings": "96",
"entrydata": [
{
"@columnnumber": "0",
"@name": "$64",
"number": {
"0": "0"
}
},
{
"@columnnumber": "1",
"@name": "$126",
"text": {
"0": "Berry2, Isabel"
}
},
{
"@columnnumber": "2",
"@name": "$Email",
"text": {
"0": "Isabel.D.Berry@spambob.com"
}
}
]
}
]
```

<http://host/path/db.nsf/view?ReadViewEntry&outputformat=JSON>

Ansichtsdaten als JSON

Domino Format

```
{
"@timestamp": "20140227T165433,89Z",
"@toplevelentries": "96",
"viewentry": [
{
"@position": "1",
"@unid": "C125796E00337519802572570066BA1F",
"@noteid": "9EE",
"@siblings": "96",
"entrydata": [
{
"@columnnumber": "0",
"@name": "$64",
"number": {
"0": "0"
}
},
{
"@columnnumber": "1",
"@name": "$126",
"text": {
"0": "Berry2, Isabel"
}
},
{
"@columnnumber": "2",
"@name": "$Email",
"text": {
"0": "Isabel.D.Berry@spambob.com"
}
}
]
}
]
```

<http://host/path/db.nsf/view?ReadViewEntry&outputformat=JSON>

Eigenes Format

```
{ "viewname": "People", "columncount": "17", "columnnames":
[ "$Color1", "$Color2", "$AltFullNameSort", "$SametimeInfo", "$64", "$126", "$Email", "$19",
"CompanyName", "$54", "$39", "$20", "JobTitle", "$PhotoURL", "OfficePhoneNumber", "$60" ] }
{ "unid": "C125796E00337519802572570066BA1F", "values": [ "", "", "Berry2 ,
Isabel", "Isabel.D.Berry@spambob.com", "0.0", "Berry2,
Isabel", "Isabel.D.Berry@spambob.com", "", "", "2014-02-27 17:55:26", "", "Caradon Hill
54##YNSEP##TYN-Y-CEFN, LL21##YNSEP##GB", "", "", "", "" ] }
{ "unid": "C125796E003375198025725700669D6C", "values": [ "", "", "Blackburn ,
Toby", "Toby.E.Blackburn@dodgeit.com", "0.0", "Blackburn,
Toby", "Toby.E.Blackburn@dodgeit.com", "", "", "2009-10-23 17:55:26", "", "Grenoble Road
32##YNSEP##BREDON, GL20##YNSEP##GB", "", "", "", "" ] }
{ "unid": "C125796E00337519802572580072A534", "values": [ "", "", "Booth ,
Loyd", "sanctus@inter.com", "0.0", "Booth, Loyd", "sanctus@inter.com", "", "", "2009-10-23
17:55:26", "", "P.O. Box 152, 671 Ron Filbee Ave.##YNSEP##KB71
4LJ2##YNSEP##Moldova", "", "", "", "" ] }
{ "unid": "C125796E00337519802572570066C41E", "values": [ "", "", "Brady ,
Joseph", "Joseph.G.Brady@pookmail.com", "0.0", "Brady,
Joseph", "Joseph.G.Brady@pookmail.com", "", "", "2009-10-23 17:55:26", "", "South Crescent
33##YNSEP##MAER, OST5##YNSEP##GB", "", "", "", "" ] }
{ "unid": "C125796E00337519802572570066C348", "values": [ "", "", "Brooks ,
Jasmine", "Jasmine.P.Brooks@pookmail.com", "0.0", "Brooks,
Jasmine", "Jasmine.P.Brooks@pookmail.com", "", "", "2009-10-23
17:55:26", "", "", "", "", "", "" ] }
{ "unid": "C125796E0033751980257257006507DC", "values": [ "", "", "Buckley ,
Tia", "Tia.A.Buckley@pookmail.com", "0.0", "Buckley,
Tia", "Tia.A.Buckley@pookmail.com", "", "", "2009-10-23 17:55:26", "", "Redcliffe Way
94##YNSEP##WOODHAM, KT15##YNSEP##GB", "", "", "", "" ] }
```

Mit einer XPage generiert

Ansichtsdaten als JSON

```
1 var xhr = Ti.Network.createHttpClient();
2
3 xhr.onload = function() {
4     var xhrData = this.responseText;
5     data = JSON.parse(xhrData);
6     Ti.API.log("Ansicht: " + data.viewname);
7     Ti.API.log("Spalten: " + data.columncount);
8 };
9
10 // oder: xhr.open("GET", "http://host/path/db.nsf/
11 // meineAnsicht?ReadViewEntries&outputformat=JSON");
12 xhr.open("GET", "http://host/path/db.nsf/meineXPage.xsp?view=meineAnsicht");
```

JSON Daten
stehen als
Objekt zur
Verfügung.

Lesen von JSON via HTTP, Arbeit mit JSON Daten

Dies ist nur ein Beispiel!

Schritt 2: Daten speichern

Was haben wir?

- Domino:
 - Ansichten: strukturiert, Zeilen und Spalten
Zugriff auf Daten via Lookups in sortieren Spalten
 - Dokumente: unstrukturiert,
Anzahl Items nicht vorhersehbar,
Typ der Items nicht vorhersehbar
Zugriff via UniversalID (UNID)

Was haben wir?

- Domino:
 - Ansichten: strukturiert, Zeilen und Spalten
Zugriff auf Daten via Lookups in sortieren Spalten
 - Dokumente: unstrukturiert,
Anzahl Items nicht vorhersehbar,
Typ der Items nicht vorhersehbar
Zugriff via UniversalID (UNID)
- Endgerät:
 - Lokale SQL-Datenbank (SQLite):
zwingend strukturiert
Struktur zur Laufzeit nur eingeschränkt änderbar
Zugriff auf Daten via SQL

Was haben wir?

- Domino:
 - Ansichten: strukturiert, Zeilen und Spalten
Zugriff auf Daten via Lookups in sortieren Spalten
 - Dokumente: unstrukturiert,
Anzahl Items nicht vorhersehbar,
Typ der Items nicht vorhersehbar
Zugriff via UniversalID (UNID)
- Endgerät:
 - Lokale SQL-Datenbank (SQLite):
zwingend strukturiert
Struktur zur Laufzeit nur eingeschränkt änderbar
Zugriff auf Daten via SQL

Und wie passt das zusammen?

Notes Ansicht

| col:cc | AltFullNameSort | SametimeInfo | Name | E-mail |
|--------|-----------------|-------------------|-----------------|-------------------------------|
| | Berry2, Isabel | Isabel.D.Berry@sp | Berry2, Isabel | Isabel.D.Berry@spambob.com |
| | Blackburn, Toby | Toby.E.Blackburn@ | Blackburn, Toby | Toby.E.Blackburn@dodgeit.com |
| | Booth, Loyd | sanctus@inter.com | Booth, Loyd | sanctus@inter.com |
| | Brady, Joseph | Joseph.G.Brady@ | Brady, Joseph | Joseph.G.Brady@pookmail.com |
| | Brooks, Jasmine | Jasmine.P.Brooks | Brooks, Jasmine | Jasmine.P.Brooks@pookmail.com |
| | Buckley, Tia | Tia.A.Buckley@pc | Buckley, Tia | Tia.A.Buckley@pookmail.com |
| | Burrows, Jodie | Jodie.T.Burrows@ | Burrows, Jodie | Jodie.T.Burrows@pookmail.com |

Ansichten? Einfach.

Beides strukturiert in Zeilen und Spalten.

Notes Ansicht

| col | cc | AltFullNameSort | SametimeInfo | Name | E-mail |
|-----|----|------------------|-------------------|-----------------|-------------------------------|
| | | Berry2 , Isabel | Isabel.D.Berry@sp | Berry2, Isabel | Isabel.D.Berry@spambob.com |
| | | Blackburn , Toby | Toby.E.Blackburn@ | Blackburn, Toby | Toby.E.Blackburn@dodgeit.com |
| | | Booth , Loyd | sanctus@inter.com | Booth, Loyd | sanctus@inter.com |
| | | Brady , Joseph | Joseph.G.Brady@ | Brady, Joseph | Joseph.G.Brady@pookmail.com |
| | | Brooks , Jasmine | Jasmine.P.Brooks | Brooks, Jasmine | Jasmine.P.Brooks@pookmail.com |
| | | Buckley , Tia | Tia.A.Buckley@pc | Buckley, Tia | Tia.A.Buckley@pookmail.com |
| | | Burrows , Jodie | Jodie.T.Burrows@ | Burrows, Jodie | Jodie.T.Burrows@pookmail.com |

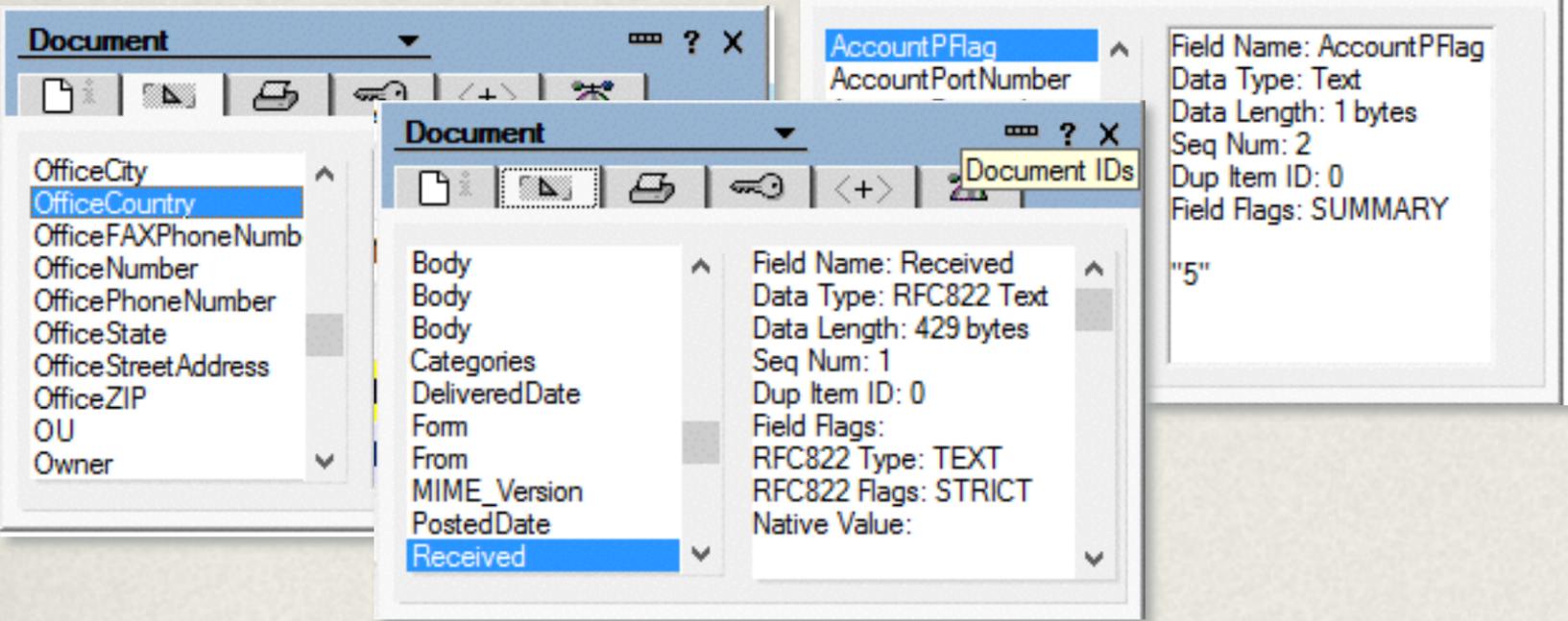
SQLite Tabelle

| | id | _altfullnamesort | _sametimeinfo | _64 | _126 | _email |
|----|--------------|------------------|------------------|-----|------------------|------------------|
| 1 | C125796E0033 | Berry2 , Isabel | Isabel.D.Berry@ | 0.0 | Berry2, Isabel | Isabel.D.Berry@ |
| 2 | C125796E0033 | Blackburn , Tob | Toby.E.Blackbu | 0.0 | Blackburn, Tob | Toby.E.Blackbu |
| 3 | C125796E0033 | Booth , Loyd | sanctus@inter.c | 0.0 | Booth, Loyd | sanctus@inter.c |
| 4 | C125796E0033 | Brady , Joseph | Joseph.G.Brady | 0.0 | Brady, Joseph | Joseph.G.Brady |
| 5 | C125796E0033 | Brooks , Jasmin | Jasmine.P.Brook | 0.0 | Brooks, Jasmine | Jasmine.P.Brook |
| 6 | C125796E0033 | Buckley , Tia | Tia.A.Buckley@ | 0.0 | Buckley, Tia | Tia.A.Buckley@ |
| 7 | C125796E0033 | Burrows , Jodie | Jodie.T.Burrows | 0.0 | Burrows, Jodie | Jodie.T.Burrows |
| 8 | C125796E0033 | Calvert , Courtr | quidem.prae@n | 0.0 | Calvert, Courtn | quidem.prae@n |
| 9 | C125796E0033 | Carrier , Brande | tempero@milie: | 0.0 | Carrier, Brander | tempero@milie: |
| 10 | C125796E0033 | Chamberlain , M | Mia.Z.Chamberl | 0.0 | Chamberlain, M | Mia.Z.Chamberl |
| 11 | C125796E0033 | Childress , Fore | mereo.defleo@c | 0.0 | Childress, Fore: | mereo.defleo@c |
| 12 | C125796E0033 | Cline , Trevor | presto.propono | 0.0 | Cline, Trevor | presto.propono |
| 13 | C125796E0033 | Collins , Randy | ledo.faveo@pro | 0.0 | Collins, Randy | ledo.faveo@pro |
| 14 | C125796E0033 | Coon , Fidel | eripio@poposco | 0.0 | Coon, Fidel | eripio@poposco |
| 15 | C125796E0033 | Cramer , Timotl | secretum.secul | 0.0 | Cramer, Timoth | secretum.secul |
| 16 | C125796E0033 | Currie , Roger | dispono@prolat | 0.0 | Currie, Roger | dispono@prolat |
| 17 | C125796E0033 | Davidson , Mas | Mason.F.Davids | 0.0 | Davidson, Masc | Mason.F.Davids |
| 18 | C125796E0033 | Deleon , Terenc | investigo.ille@l | 0.0 | Deleon, Terenc | investigo.ille@l |
| 19 | C125796E0033 | Dodd , Elizabet | Elizabeth.A.Doc | 0.0 | Dodd, Elizabeth | Elizabeth.A.Doc |
| 20 | C125796E0033 | Dodd , Elizabet | Elizabeth.A.Doc | 0.0 | Dodd, Elizabeth | Elizabeth.A.Doc |

Ansichten? Einfach.

Beides strukturiert in Zeilen und Spalten.

Notes Dokumente



SQLite Tabelle

| | id | _altfullnamesort | _sametimeinfo | _64 | _126 | _email |
|----|--------------|------------------|------------------|-----|------------------|------------------|
| 1 | C125796E0033 | Berry2 , Isabel | Isabel.D.Berry@ | 0.0 | Berry2, Isabel | Isabel.D.Berry@ |
| 2 | C125796E0033 | Blackburn , Tob | Toby.E.Blackbu | 0.0 | Blackburn, Tob | Toby.E.Blackbu |
| 3 | C125796E0033 | Booth , Loyd | sanctus@inter.c | 0.0 | Booth, Loyd | sanctus@inter.c |
| 4 | C125796E0033 | Brady , Joseph | Joseph.G.Brady(| 0.0 | Brady, Joseph | Joseph.G.Brady(|
| 5 | C125796E0033 | Brooks , Jasmin | Jasmine.P.Brook | 0.0 | Brooks, Jasmin | Jasmine.P.Brook |
| 6 | C125796E0033 | Buckley , Tia | Tia.A.Buckley@ | 0.0 | Buckley, Tia | Tia.A.Buckley@ |
| 7 | C125796E0033 | Burrows , Jodie | Jodie.T.Burrows | 0.0 | Burrows, Jodie | Jodie.T.Burrows |
| 8 | C125796E0033 | Calvert , Courtr | quidem.prae@n | 0.0 | Calvert, Courtn | quidem.prae@n |
| 9 | C125796E0033 | Carrier , Brande | tempero@milie: | 0.0 | Carrier, Brander | tempero@milie: |
| 10 | C125796E0033 | Chamberlain , M | Mia.Z.Chamberl | 0.0 | Chamberlain, M | Mia.Z.Chamberl |
| 11 | C125796E0033 | Childress , Fore | mereo.defleo@c | 0.0 | Childress, Fore | mereo.defleo@c |
| 12 | C125796E0033 | Cline , Trevor | presto.propono | 0.0 | Cline, Trevor | presto.propono |
| 13 | C125796E0033 | Collins , Randy | ledo.faveo@pro | 0.0 | Collins, Randy | ledo.faveo@pro |
| 14 | C125796E0033 | Coon , Fidel | eripio@poposco | 0.0 | Coon, Fidel | eripio@poposco |
| 15 | C125796E0033 | Cramer , Timot | secretum.secul | 0.0 | Cramer, Timoth | secretum.secul |
| 16 | C125796E0033 | Currie , Roger | dispono@prolat | 0.0 | Currie, Roger | dispono@prolat |
| 17 | C125796E0033 | Davidson , Mas | Mason.F.Davids | 0.0 | Davidson, Masc | Mason.F.Davids |
| 18 | C125796E0033 | Deleon , Terenc | investigo.ille@l | 0.0 | Deleon, Terenc | investigo.ille@l |
| 19 | C125796E0033 | Dodd , Elizabet | Elizabeth.A.Doc | 0.0 | Dodd, Elizabet | Elizabeth.A.Doc |
| 20 | C125796E0033 | Duda , Chadia | Chadial.Duda | 0.0 | Duda, Chadia | Chadial.Duda |

Dokumente? Schwierig.

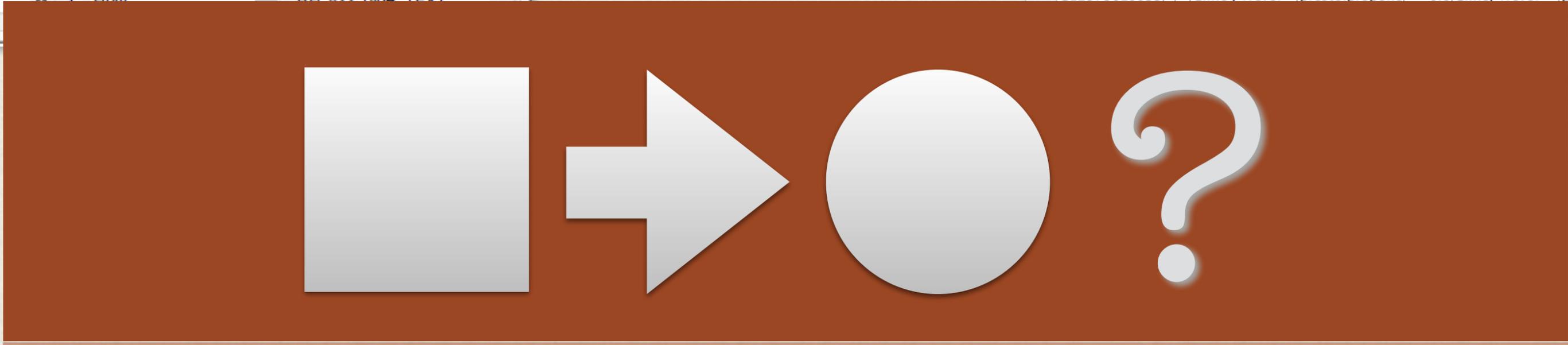
Notes Dokumente

AccountPFlag
Data Type: Text
Data Length: 1 bytes
Seq Num: 2
Dup Item ID: 0
Field Flags: SUMMARY
"5"

Received
Data Type: RFC822 Text
Data Length: 429 bytes
Seq Num: 1
Dup Item ID: 0
Field Flags: RFC822 Type: TEXT

SQLite Tabelle

| | id | _altfullnamesort | _sametimeinfo | _64 | _126 | _email |
|----|--------------|------------------|-----------------|-----|------------------|-----------------|
| 1 | C125796E0033 | Berry2 , Isabel | Isabel.D.Berry@ | 0.0 | Berry2, Isabel | Isabel.D.Berry@ |
| 2 | C125796E0033 | Blackburn , Tob | Toby.E.Blackbu | 0.0 | Blackburn, Tob | Toby.E.Blackbu |
| 3 | C125796E0033 | Booth , Loyd | sanctus@inter.c | 0.0 | Booth, Loyd | sanctus@inter.c |
| 4 | C125796E0033 | Brady , Joseph | Joseph.G.Brady@ | 0.0 | Brady, Joseph | Joseph.G.Brady@ |
| 5 | C125796E0033 | Brooks , Jasmin | Jasmine.P.Brook | 0.0 | Brooks, Jasmin | Jasmine.P.Brook |
| 6 | C125796E0033 | Buckley , Tia | Tia.A.Buckley@ | 0.0 | Buckley, Tia | Tia.A.Buckley@ |
| 7 | C125796E0033 | Burrows , Jodie | Jodie.T.Burrows | 0.0 | Burrows, Jodie | Jodie.T.Burrows |
| 8 | C125796E0033 | Calvert , Courtr | quidem.prae@n | 0.0 | Calvert, Courtr | quidem.prae@n |
| 9 | C125796E0033 | Carrier , Brande | tempero@milie: | 0.0 | Carrier, Brander | tempero@milie: |
| 10 | C125796E0033 | Chamberlain , M | Mia.Z.Chamberl | 0.0 | Chamberlain, M | Mia.Z.Chamberl |
| 11 | C125796E0033 | Childress , Fore | mereo.defleo@c | 0.0 | Childress, Fore | mereo.defleo@c |
| 12 | C125796E0033 | Cline , Trevor | presto.propono | 0.0 | Cline, Trevor | presto.propono |



Dokumente? Schwierig.

Lösung: XML

Notes XML (DXL) mit UNID als primary key in SQL Tabelle speichern.

Notes Dokument als XML

```
<?xml version='1.0'?>
<!DOCTYPE document SYSTEM 'xmlschemas/domino_9_0_1.dtd'>
<document xmlns='http://www.lotus.com/dxl' version='9.0' maintenanceversion='1.0'
replicaid='C125796E00330517' form='Person'>
<noteinfo noteid='9ee' unid='C125796E00337519802572570066BA1F' sequence='6'>
<created><datetime>20070102T184203,19+00</datetime></created>
<modified><datetime>20140227T092020,82+01</datetime></modified>
<revised><datetime>20140227T092020,80+01</datetime></revised>
<lastaccessed><datetime>20140227T092020,80+01</datetime></lastaccessed>
<addedtofile><datetime>20111222T104354,49+01</datetime></addedtofile></note
<updatedby><name>CN=Jake Howlett/O=ROCKALL</name><name>CN=Julian
Buss/O=YouAtNotes</name><name
>Anonymous</name><name>CN=Julian
Buss/O=YouAtNotes</name><name>Anonymous</name></updatedby>
<revisions><datetime>20070102T184203,21+00</datetime><datetime
dst='true'>20091023T102908,53+02</datetime><datetime
dst='true'>20130703T170246,14+02</datetime><datetime>20140227T083856,97+0
>20140227T083857,30+01</datetime></revisions>
<item name='SEX'><text>female</text></item>
<item name='OfficeNumber'><number>769214</number></item>
<item name='LASTMOD'><datetime>19560202T000000,00+00</datetime></item>
<item name='preferredLanguage'><text/></item>
<item name='ccMailLocation'><text/></item>
```

Lösung: XML

Notes XML (DXL) mit UNID als primary key in SQL Tabelle speichern.

Notes Dokument als XML

```
<?xml version='1.0'?>
<!DOCTYPE document SYSTEM 'xmlschemas/domino_9_0_1.dtd'>
<document xmlns='http://www.lotus.com/dxl' version='9.0' maintenanceversion='1.0'
replicaid='C125796E00330517' form='Person'>
<noteinfo noteid='9ee' unid='C125796E00337519802572570066BA1F' sequence='6'>
<created><datetime>20070102T184203,19+00</datetime></created>
<modified><datetime>20140227T092020,82+01</datetime></modified>
<revised><datetime>20140227T092020,80+01</datetime></revised>
<lastaccessed><datetime>20140227T092020,80+01</datetime></lastaccessed>
<addedtofile><datetime>20111222T104354,49+01</datetime></addedtofile></note
<updatedby><name>CN=Jake Howlett/O=ROCKALL</name><name>CN=Julian
Buss/O=YouAtNotes</name><name
>Anonymous</name><name>CN=Julian
Buss/O=YouAtNotes</name><name>Anonymous</name></updatedby>
<revisions><datetime>20070102T184203,21+00</datetime><datetime
dst='true'>20091023T102908,53+02</datetime><datetime
dst='true'>20130703T170246,14+02</datetime><datetime>20140227T083856,97+0
>20140227T083857,30+01</datetime></revisions>
<item name='SEX'><text>female</text></item>
<item name='OfficeNumber'><number>769214</number></item>
<item name='LASTMOD'><datetime>19560202T000000,00+00</datetime></item>
<item name='preferredLanguage'><text/></item>
<item name='ccMailLocation'><text/></item>
```

SQLite Tabelle

| | | xml |
|---|-------------|--------------------------|
| 1 | 2570066BA1F | <?xml version='1.0'?>... |
| 2 | 25700669D6C | <?xml version='1.0'?>... |
| 3 | 2580072A534 | <?xml version='1.0'?>... |
| 4 | 2570066C41E | <?xml version='1.0'?>... |
| 5 | 2570066C348 | <?xml version='1.0'?>... |
| 6 | 257006507DC | <?xml version='1.0'?>... |
| 7 | 25700652295 | <?xml version='1.0'?>... |

Lösung: XML

Notes XML (DXL) mit UNID als primary key in SQL Tabelle speichern.

Was ist mit RichText?

Was ist mit RichText?

- Es gibt keine Möglichkeit, Notes-RichText so wie es ist auf dem Endgerät darzustellen.
- Ausweg: HTML
- Mittels XPage HTML Repräsentation von RichText erstellen und via HTTP an die App übertragen.
- App hat ein Control zum anzeigen von HTML.
- Schwierig: Bilder.

Was ist mit RichText?

- Es gibt keine Möglichkeit, Notes-RichText so wie es ist auf dem Endgerät darzustellen.
- Ausweg: HTML
- Mittels XPage HTML Repräsentation von RichText erstellen und via HTTP an die App übertragen.
- App hat ein Control zum anzeigen von HTML.
- Schwierig: Bilder.

Beispiel-Code in Java:

```
doc.convertToMIME(2);  
MIMEEntity mime = doc.getMIMEEntity("Body");  
mime = mime.getFirstChildEntity();  
mime.decodeContent();  
String html = mime.getContentAsText();
```

RichText in Notes und in der App

The screenshot shows the IBM Notes desktop application window titled "interesting links for XPages development - IBM Notes". The interface includes a menu bar (File, Edit, View, Create, Actions, Tools, Window, Help), a toolbar with "Open", "Forward", and "Print Without Header" options, and a list of open documents. The main content area displays a note with the following details:

- Subject:** interesting links for XPages development
- Entry Date:** So 04.12.2011
- Modified:** 25.04.2012
- Category:** work, development

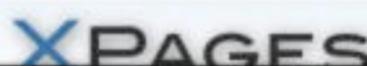
The note content is structured as follows:

| | |
|---|--|
|  | xpageswiki.com This site provides a lot of howtos, tips and ready-to-use code about XPages development on IBM Lotus Domino or the IBM XWork server. It is maintained by Julian Buss from YouAtNotes . |
|  | xpages.info Rapid Web and Mobile Application Development Platform |
| | CheatSheet |
| | PDF Attachment: XPagesCheatSheet-11x17-v1.pdf |

The screenshot shows the IBM Notes mobile application interface. The status bar at the top displays "Carrier", signal strength, Wi-Fi, and the time "6:10 PM". The note header shows the subject "interesting links..." and a pencil icon for editing. The note content is displayed as follows:

- Subject:** interesting links for XPages development
- Category:** work, development
- Entry Date:** 12/04/2011

The note content is structured as follows:

| | |
|---|--|
|  | xpageswiki.com This site provides a lot of howtos, tips and ready-to-use code about XPages development on IBM Lotus Domino or the IBM XWork server. It is maintained by Julian Buss from YouAtNotes . |
|  | xpages.info |

The bottom of the screen features a navigation bar with icons for "Subject", "Date", "Categories", "Settings", and "Help".

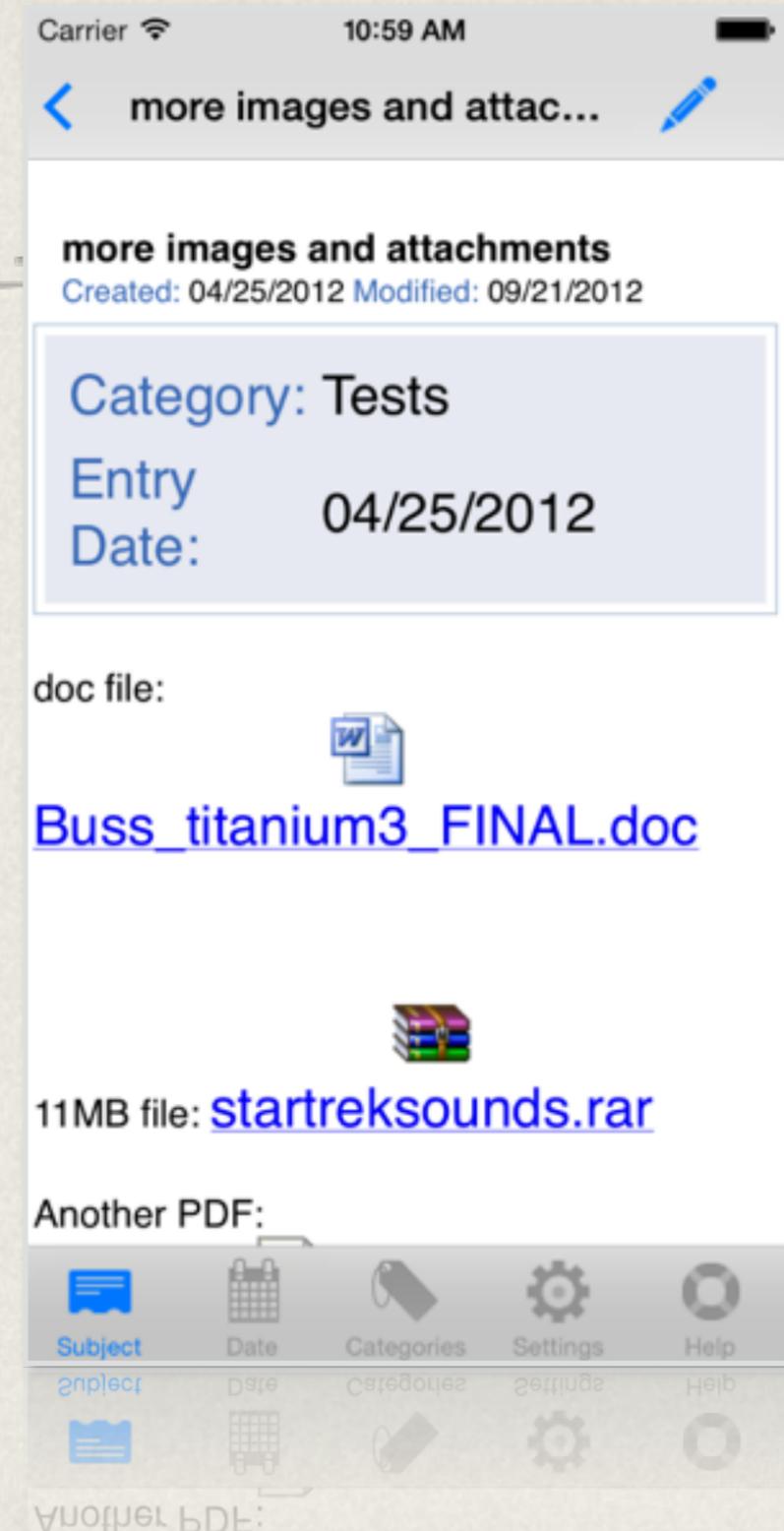
Anhänge

Anhänge

- In XML vom Notes Dokument prüfen, ob Anhänge existieren
- Wenn ja, Dateinamen aus XML extrahieren
- Dann via HTTP herunterladen und im lokalen Dateisystem auf dem Endgerät speichern
- Auf dem Endgerät dann die nativen Möglichkeiten nutzen (Datei darstellen, senden...)

Anhänge

- In XML vom Notes Dokument prüfen, ob Anhänge existieren
- Wenn ja, Dateinamen aus XML extrahieren
- Dann via HTTP herunterladen und im lokalen Dateisystem auf dem Endgerät speichern
- Auf dem Endgerät dann die nativen Möglichkeiten nutzen (Datei darstellen, senden...)



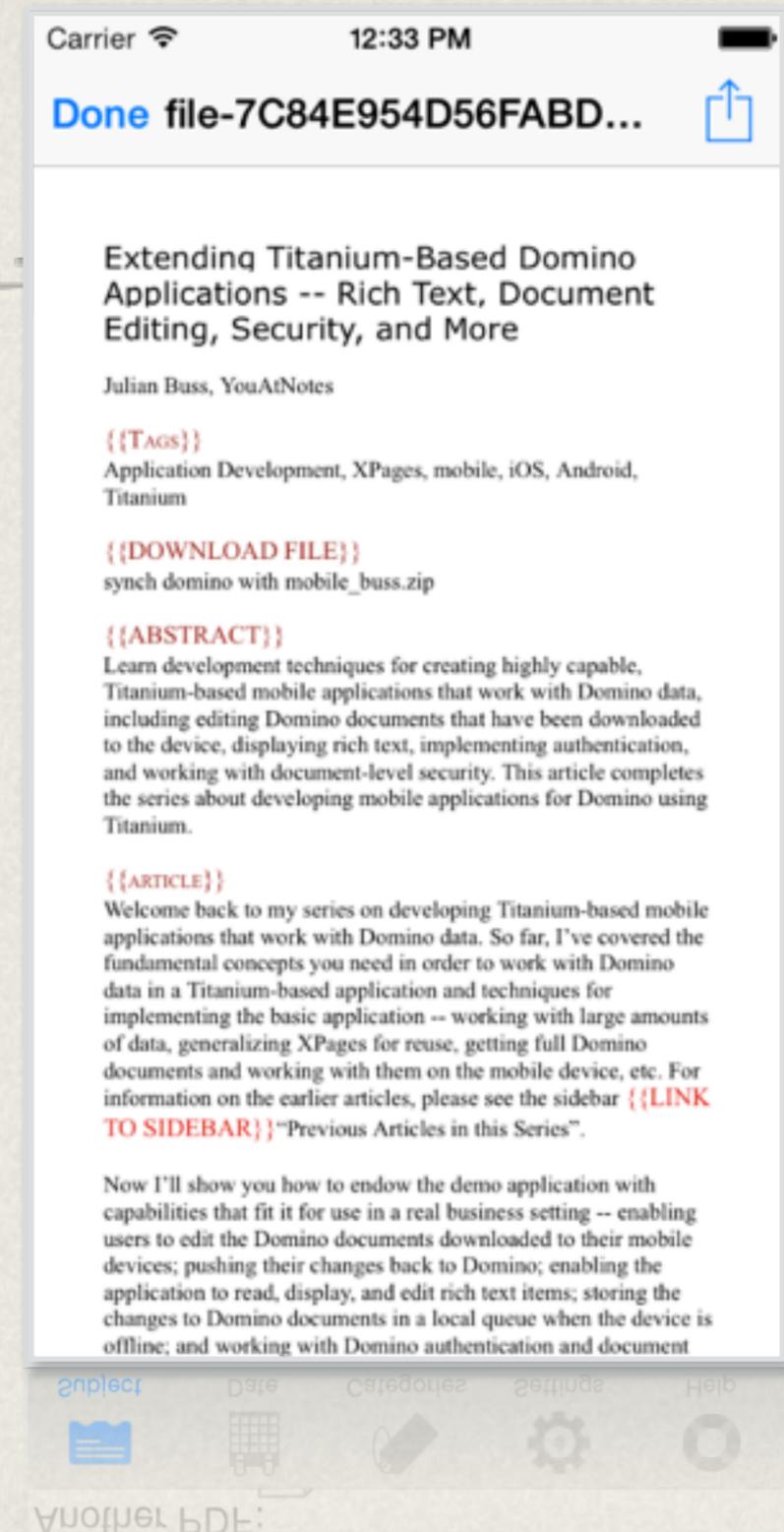
Anhänge

- In XML vom Notes Dokument prüfen, ob Anhänge existieren
- Wenn ja, Dateinamen aus XML extrahieren
- Dann via HTTP herunterladen und im lokalen Dateisystem auf dem Endgerät speichern
- Auf dem Endgerät dann die nativen Möglichkeiten nutzen (Datei darstellen, senden...)



Anhänge

- In XML vom Notes Dokument prüfen, ob Anhänge existieren
- Wenn ja, Dateinamen aus XML extrahieren
- Dann via HTTP herunterladen und im lokalen Dateisystem auf dem Endgerät speichern
- Auf dem Endgerät dann die nativen Möglichkeiten nutzen (Datei darstellen, senden...)



Bis hierin geschafft:

- Daten lesen:
 - Netz: Internet, ggf. VPN
 - Protokoll: HTTP bzw. HTTPS
 - Format: JSON oder XML
ggf. eigenes JSON, da Domino JSON viel Ballast enthält
 - JSON und XML können in JavaScript im Titanium einfach verarbeitet werden
- Daten speichern:
 - Ansicht: direkt in SQL Tabelle
 - Dokumente: als XML in SQL Tabelle mit UNID als Key
 - RichText: als HTML
 - Anhänge: in lokalem Dateisystem

Schritt 3: Daten verarbeiten

Möglichkeit 1

Standard Titanium Code

```
1 var db = Ti.Database.open("contacts");
2 var dbRows = db.execute("SELECT * FROM view_people");
3 while (dbRows && dbRows.isValidRow()) {
4
5     var tableRow = {
6         id: dbRows.fieldByName("id"),
7         title : dbRows.fieldByName("name")+", "+dbRows.fieldByName("email")
8     };
9     tableData.push(tableRow);
10
11     dbRows.next();
12 }
13 dbRows.close();
14 db.close();
15 table.setData(tableData);
```

Möglichkeit 2 und
mehr dazu in
Session 2!

Schritt 4: Daten hochladen

Änderungen an Dokumenten

- Items in einem Dokument auf dem Endgerät ändern:
 - lokale XML Repräsentation ändern und in SQLite speichern
 - XML zu Domino hochladen?
 - Nein: zu viel Ballast!
 - Besser: nur Namen und Inhalte der geänderten Items in eigenem JSON Format übertragen

JavaScript Objekt via HTTP an Domino senden (Beispiel)

```
1 var data = {};  
2 data.items = [];  
3 data.items.push(  
4     uid: "7c84e954d56fabd4c12579eb004c9bc0",  
5     itemname: "lastname",  
6     itemvalue: "sisko"  
7 });  
8 data.items.push(  
9     uid: "7c84e954d56fabd4c12579eb004c9bc0",  
10    itemname: "firstname",  
11    itemvalue: "benjamin"  
12 });  
13  
14 var xhr = Ti.Network.createHttpClient();  
15 xhr.onload = function() {  
16     var data = this.responseText;  
17     if (data != "OK") {  
18         Ti.API.log("Upload fehlgeschlagen.");  
19     }  
20 };  
21 xhr.open("POST", "http://host/path/db.nsf/write.xsp");  
22 xhr.send(JSON.stringify(data));
```

In Titanium:
JavaScript Objekt mit
Daten erzeugen,
HTTP Client holen,
JavaScript Objekt über
HTTP an eine XPage
senden.

JavaScript Objekt via HTTP an Domino senden (Beispiel)

```
var data = param.toString();
data = fromJson(data);
for (var i = 0; i < data.items.length; i++) {
    var doc = database.getDocument(data.items[i].unid);
    doc.replaceItemValue(data.items[i].itemname, data.items[i].itemvalue);
    doc.save();
}

var response = exCon.getResponse();
response.setContentType("text/plain");
response.setHeader("Cache-Control", "no-cache");
writer.write("OK");
writer.endDocument();
facesContext.responseComplete();
```

In Domino XPage:
Daten aus HTTP Request
holen,
JavaScript Objekt aus JSON
Daten erzeugen,
Items-Array durchgehen,
Dokument holen und Item-
Wert ändern,
„OK“ zurückgeben

Was ist mit Anhängen?

- Geht, ist aber etwas anstrengender:
 - In Titanium: Datei mit base64 zu Text wandeln und hochladen.
 - Auf XPage Seite: base64 Daten wieder decodieren, passendes MIME Entity im Dokument erzeugen und Daten dort reinschreiben.
 - Wenn im Dokument bisher kein MIME Item besteht, muss es vorher erzeugt werden.
 - Bei mehreren Anhängen entsprechend mehrere MIME Entities erzeugen und an das Multipart MIME Item dranhängen.

Was ist mit RichText

- Unwichtig, da RichText auf dem Endgerät in der Regel sowieso nicht bearbeitet werden kann.
- Wenn doch wichtig: RichText als HTML zur Domino XPage hochladen und dann wieder ein passendes MIME Entity im Dokument erzeugen.

Fertig! Wir können:

- Daten lesen:
 - Netz: Internet, ggf. VPN
 - Protokoll: HTTP bzw. HTTPS
 - Format: JSON oder XML
ggf. eigenes JSON, da Domino JSON viel Ballast enthält
 - JSON und XML können in JavaScript im Titanium einfach verarbeitet werden
- Daten speichern:
 - Ansicht: direkt in SQL Tabelle
 - Dokumente: als XML in SQL Tabelle mit UNID als Key
 - RichText: als HTML
 - Anhänge: in lokalem Dateisystem

...und:

- Daten verarbeiten:
 - z.B. mit Standard Titanium API
Ansichtsdaten lesen und darstellen
 - oder Möglichkeit 2, die wir erst
in Session 2 sehen :-)
- Daten hochladen:
 - Daten in JavaScript Objekte
verpacken und als JSON
hochladen
 - In Domino diese JSON Daten mit
XPage empfangen und
verarbeiten

Security

- Netz & Transport:
 - Transportverschlüsselung mit SSL
 - bei Bedarf VPN
- Auf dem Gerät:
 - lokale Verschlüsselung z.B. bei iOS
 - ggf. Verpacken in Enterprise-Container bei passendem MDM System
- Domino:
 - App ist wie ein Web User
 - Normale Authentifizierung
 - ACL, Leser- und Autorenfelder wirken wie immer

Zwischenfazit

- mit einem Framework wie Titanium wird die Entwicklung nativer Apps wesentlich erleichtert
- Synchronisation, Speicherung und Verarbeitung von Notes-Daten in der App ist grundsätzlich machbar
- Security ist gegeben
- native Apps haben Vorteile bzgl. offline, Look & Feel

In Session 2:

Beispiele mit echtem Code,
Erfahrungen aus dem
echten Leben,
Überlegungen zur Performance

In Session 2:

Beispiele mit echtem Code,
Erfahrungen aus dem
echten Leben,
Überlegungen zur Performance

Morgen, Dienstag, 11 Uhr, Track 3!

Danke! Fragen?



Julian Buß
<http://julianbuss.net>
jbuss@julianbuss.net

Mein Framework für Sync & Verarbeitung von
Notes-Daten in Titanium:
<http://youatnotes.com/dominotogo>

Titanium gibt es kostenlos unter:
<http://appcelerator.com>

Blog-Einträge zum Thema:
<http://julianbuss.com/blog/category/dominotogo/>