



Web2.0的地図/スケジュール 共有サービスの実装

XMLコンソーシアムWeek 関西Day 2006/06/20

XMLコンソーシアム 関西部会

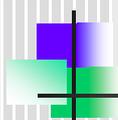
コベルコシステム株式会社 幸田 健

Sky株式会社 玉川 竜司

日本アイ・ビー・エム株式会社 太田 一郎

株式会社プレイネットワークス 丸山 和秀

株式会社プレイネットワークス 中川 聖悠



アジェンダ

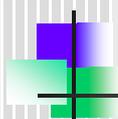


- 概要
- デモ
- 認証実装
- Webサービス実装
- クライアント実装
- まとめ



概要

コベルシステム株式会社 幸田 健



背景・目的



- 実際の開発を経験することにより、トレンド技術の習得とノウハウの獲得を行う
 - 興味のある新技術に挑戦する
 - 自分たちが使いたいものをつくる
 - いつでも参加可能

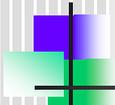


プロジェクトの変遷



- 開発プロジェクトの段階
 - ステップ1(2005年1月～5月):
複数のWebサービスを利用したアプリケーションの開発
 - スケジュールサービス、居場所サービス、天気情報サービスの開発
 - リッチクライアント(.NETクライアント、Flash)の開発
 - ステップ2 (2005年6月～2006年5月):
Web 2.0的アプリケーションのスタイルを取り入れ再設計
 - 地図API(Google Maps API)の活用
 - スケジュールサービス、場所情報サービス、認証サービスの開発
 - ブラウザのみで動くリッチなクライアント(AJAX)
 - 認証機能の追加

AJAX:Asynchronous JavaScript And XML



関西部会におけるWeb2.0とは [マインド編]



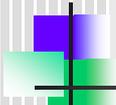
- 参加型
 - アプリケーションの設計・開発プロセス
 - 参加型アプリケーション
- 永遠にベータ版
 - DIY (Do IT Yourself)マインド
 - リリースを決めずいつもバージョンアップ
- リミックス (マッシュアップ)
 - 「いちからつくる」のではなく、「組み合わせでつくる」
 - 必要に応じて複数のWebサービスを組み合わせる



関西部会におけるWeb2.0とは [実装技術編]



- リッチなユーザインタフェース
 - 「利用者を待たせない」インタフェース
 - AJAXの利用
 - 情報の「見える化」の促進
 - タグクラウド(キーワードクラウド)の利用
- ブラウザさえあれば動く
 - プラットフォームとしてのブラウザ (Browser as a platform)
 - XHTML + JavaScript + CSS + Webサービス
- 軽量のプログラミングモデル
 - Webサービス
 - XMLを使ったりリモート呼び出し・データ転送 (SOAP、REST API、XML-RPC の利用)
 - スクリプティング言語
 - JavaScript、Pythonなどの利用



アプリケーションの要件



- 実装するアプリケーションの要件
 - 自分のスケジュールがわかる
 - 仲間のスケジュールがわかる
 - 地図上に登録した場所情報が表示される
 - スケジュールから居場所がわかる
 - 場所付近の天気がわかる (未実装)

 いまどこをなにしていますのwTM

必要となるWebサービス



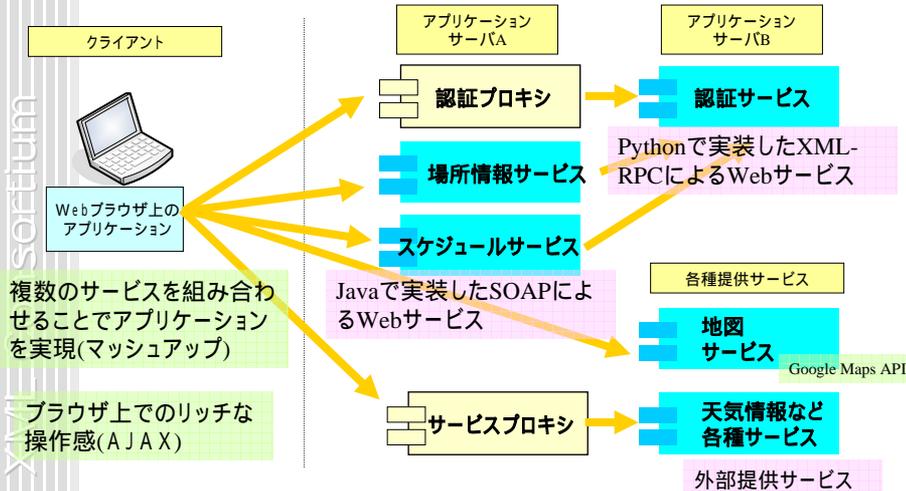
XML Consortium

- 認証サービス
 - 各サービス利用時にユーザ認証を行うためのサービス
 - アプリケーションへのログイン時に利用される
 - 各サービス(スケジュール・場所情報サービス)の利用時に参照される
- スケジュールサービス
 - ユーザのスケジュール情報の参照、登録、更新、削除するサービス
- 地図サービス
 - 指定された地域の地図を提供するサービス
(例: Google Maps API)
- 場所情報サービス
 - 地図上の場所に関するデータの参照、登録、更新、削除
- 天気情報サービス
 - 指定された地域の天気情報を提供するサービス

全体の構成 (概要)



XML Consortium



要素技術の説明



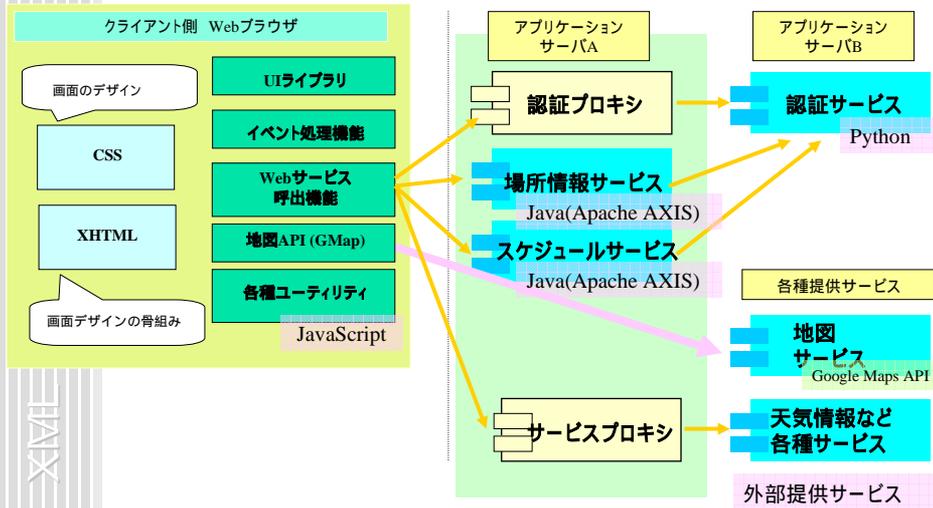
- SOAP
 - Webサービスの標準プロトコル(W3C)
 - RPCと文書通信が可能
 - NameSpaceを用いる
 - Apache AXISで実装されている
- XML-RPC
 - HTTP上のXMLを使ったりリモート呼び出し
 - シンプルなXML(Namespaceを用いない)
 - ブログの記事投稿で利用されている
- REST(Representational State Transfer) API
 - Webシステムのアーキテクチャスタイル
 - 特定のURIに対してHTTPのGET,POST,PUT,DELETEで操作する
 - RESTを満たす実装として、ATOM Publishing Protocol(IETF)がある

実装する機能



- クライアント側
 - ユーザインタフェースとそのイベント処理機能
 - 地図の操作、カレンダーの操作 など
 - Webサービス呼出機能
- サーバー側
 - Webサービスプロバイダー機能
 - 開発し提供するWebサービス(場所情報、スケジュール、認証など)
 - Webサービスの集約機能
 - 別ネットワーク・ドメインで提供されるWebサービスを仲介(ブラウザのセキュリティの制約)

全体の構成(詳細)



© XML Consortium



XML Consortium

デモ

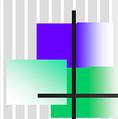
株式会社ブレインワークス 中川 聖悠

© XML Consortium



認証サービス

Sky株式会社 玉川竜司



認証サービスとは？



- ユーザアカウントを管理する
- 他のサービスからのリクエストを受け、以下の処理を行う
 - 認証処理(ユーザ名・パスワードは正しいか)
 - セッションの生成(ユーザがシステムを使い始める)
 - セッションの有無(ユーザがログイン中か)
 - セッションの廃棄(一定期間放置で強制ログアウト)

認証サービス(技術概要)



- バックエンドとしてオープンソースのXMLDBを利用してみる
- 個人情報の保存にmicroformatsを使用してみる
- APIはXMLRPCで実装

バックエンドのXMLDB



- オープンソースのデータベース・eXist
 - <http://exist.sourceforge.net/>
 - Pure Java
 - 完全にネイティブのXMLDB
 - 現在もアクティブに開発進行中
 - XPath / XQuery
 - XMLRPC / SOAP / RESTのAPI

なぜXMLDBを使ったか？

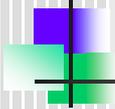


- パフォーマンスや運用面ではRDBのほうが優れている(少なくとも現時点では)
 - 単なる認証サービスだけならRDBで十分
- microformatsのデータをそのまま保存できる
 - 単なる認証情報だけでなく、他の情報のDBとしても活用できるのではないか(将来的に)
 - 友達情報、スケジュールなど
 - スキーマを固めなくてもいいのはXMLDBの魅力
- RDBと比較して、決定的なメリットはまだ見えない？

でも、XMLDBで大丈夫？



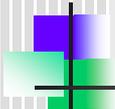
- 運用面での不安
 - これは実績がないので仕方がない面もある
 - 時間が解決する(とりたい)
- パフォーマンスの不安
 - eXistの場合
 - データの検索は意外に速い
 - データの挿入も速い
 - 削除は遅い...



microformats



- <http://microformats.org/>
- さまざまな種類のデータを、ごく簡単なXMLで定義している
- 100%を目指す、数年後に実用化されるフォーマットよりは、80%でいいから今すぐ使えるフォーマット
- 仕様は極力シンプル、すぐにいじれる
- バックエンドのデータにCSSやXQuery / XSLTを適用すれば、そのまま表示に利用できる
- 今回の実装では手始めとしてhCardを使用
 - 次はXFNを使っていきたい(友達検索)



サーバの実装とAPI



- XMLRPCベースのAPIを実装
 - SOAP / RESTも実装予定...
- Pythonで実装(Fedora Core 3上で稼動)
- eXistともXMLRPCで通信
- セキュリティの問題には今回は目をつぶっている(今後のテーマ)

今後のテーマ



- サーバ間の認証情報交換プロトコルへの対応: Sxipなど
- XMLDBの利便性の検証
 - より広範囲なmicroformatsベースのデータの管理(友達関係、スケジュールなど)
 - XQueryをどんどん使ってみる
- XMLDBのパフォーマンス検証
 - 定量的な検証を行いたい



Webサービス実装

日本アイ・ビー・エム株式会社 太田 一郎



Webサービス実装の概要



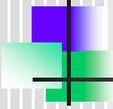
- 実装したサービス
 - 居場所サービス
 - スケジュールサービス
- 認証サービスとの連携
 - クライアント ~ 認証サービスのゲートウェイの提供
 - 保護対象 URL へのアクセスは Servlet でフィルタリング
- 開発プラットフォーム
 - Tomcat 5.5.17
 - Axis 1.3
 - DB はインメモリーのオブジェクトで代用



居場所サービス



サービス概要	地図上の場所を管理するサービス。場所名、地図上の座標 (Google Maps と連携)、キーワード、カテゴリの情報を一括して「場所」情報として管理。
提供機能	<ul style="list-style-type: none">■ 新規登録■ 更新■ 削除■ 各種検索機能<ul style="list-style-type: none">■ 場所名で■ キーワードで■ カテゴリで■ 登録者で■ 座標でetc ...



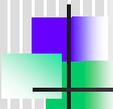
居提供サービス詳細



XML Consortium

場所サービスの検索メソッド

- Location findById(String locationId) idによる検索
- Location findByCoordinate(double latitude, double longitude) 緯度経度による検索
- Location[] findByArea(double latitude1, double longitude1, double latitude2, double longitude2) 長方形エリアの範囲内検索(長方形の対角線のみを指定)
- Location[] findByComment(String description) 説明文字列による検索
- Location[] findByAddress(String address) 住所文字列による検索
- Location[] findByCategory(String category) カテゴリによる検索
- Location[] findByKeyword(String keyword) キーワードによる検索
- Location[] findByUserId(String userId) ユーザidによる検索
- String[] findByIdByComment(String description) 説明文字列による検索
- String[] findByIdByAddress(String address) 住所文字列による検索
- String[] findByIdByCategory(String category) カテゴリによる検索
- String[] findByIdByKeyword(String keyword) キーワードによる検索
- String[] findByIdByUserId(String userId) ユーザidによる検索
- String createLocation(Location location) locationId自動生成 (Location.locationIdがnull)
- void updateLocation(Location location)
- Location deleteLocation(String locationId)
- WeightedItem[] getKeywordWithWeight() 重みつきキーワード全件取得
- WeightedItem[] getCategoryWithWeight() 重みつきカテゴリ全件取得
- String[] getKeyword() キーワード全件取得
- String[] getCategory() カテゴリ全件取得
- String[] getLocationId() locationId全件取得
- Location[] getAllLocation() Location全件取得
- WeightedItem[] findKeywordWithWeightByKeyword(String keyword) そのキーワード文字列を含む重みつきキーワードを検索



スケジュールサービス



XML Consortium

サービス概要	ユーザーの予定を管理するサービス。日時、予定の概要、登録ユーザー、場所(居場所サービスで管理されているもの)を一括して「スケジュール」情報として管理。
提供機能	<ul style="list-style-type: none"> ■ 新規登録 ■ 更新 ■ 削除 ■ 各種検索機能 <ul style="list-style-type: none"> ■ 期間で ■ 日時で ■ キーワードで ■ 登録者で etc ...



スケジュールサービス詳細



XML Consortium

予定サービスの検索メソッド

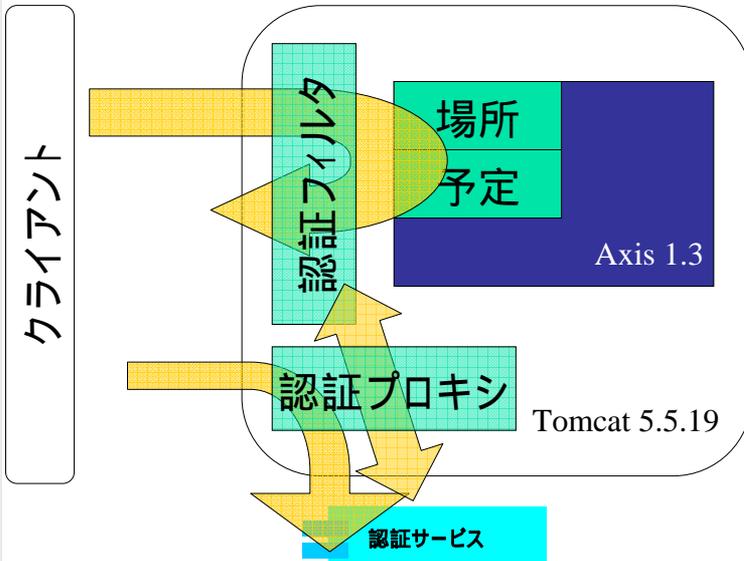
- schedule findScheduleById(String scheduleId)
- Location findLocation(String userId, DateTime datetime)
- String createSchedule(Schedule schedule)
- void updateSchedule(Schedule schedule)
- schedule deleteSchedule(String scheduleId)
- schedule[] findScheduleByMonth(int year,int month)// 月で検索
- schedule[] findScheduleByMonthAndUser(int year,int month,String userId)// 月とuserIdで検索
- schedule[] findScheduleByWeek(int year,int weekOfYear)// 週(weekOfYear)で検索
- schedule[] findScheduleByWeek(int year,int weekOfYear,String userId)// 週(weekOfYear)とuserIdで検索
- schedule[] findScheduleByDate(date date)// 日で検索
- schedule[] findScheduleByDateAndUser(date date,String userId)// 日で検索
- schedule[] findScheduleByPeriod(date date1,date date2,String userId)// 期間で検索とuserId
- String[] findScheduleByMonth(int year,int month)// 月で検索
- String[] findScheduleByMonthAndUser(int year,int month,String userId)// 月とuserIdで検索
- String[] findScheduleByWeek(int year,int weekOfYear)// 週(weekOfYear)で検索
- String[] findScheduleByWeek(int year,int weekOfYear,String userId)// 週(weekOfYear)とuserIdで検索
- String[] findScheduleByDate(date date)// 日で検索
- String[] findScheduleByDateAndUser(date date,String userId)// 日で検索
- String[] findScheduleByPeriod(date date1,date date2)// 期間で検索
- String[] findScheduleByPeriodAndUser(date date1,date date2,String userId)// 期間で検索とuserId



認証サービスの連携



XML Consortium



今後の予定



- 関西部会の共通開発プラットフォームとしての展開
 - データベースの採用 (RDBMS / XML DB)
 - 新規サービスの追加構築を容易にするための工夫

参考: データ表現



■ XMLでの表現

```
<location>
  <address>大阪市</address>
  <category>施設</category>
  <comment>集合場所</comment>
</location>
```

```
categoryデータ取得の例(DOM):
var cateData =
locationNode.childNodes[1].nodeValue
```

■ JSON(JavaScript Object Notation)での表現

```
location: { address: "大阪市", category: "施設", comment: "集合場所" }
```

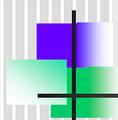
```
categoryデータ取得の例:
var cateData = location.category
```

JavaScriptからの取り扱いがシンプルに



クライアント実装

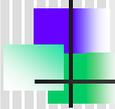
株式会社ブレインワークス 丸山 和秀



実装目標



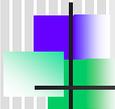
- 使えるものを目標に
 - 自分達が楽しんで使える
- すべて JavaScript で実装する
 - サーバーサイドの機能などは使わない
- JavaScript で 直接 SOAP と接続
- 使えるコンポーネントの開発
 - 使い勝手が良く再利用できる



Ajaxとは



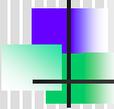
- **A**synchronous **J**avaScript and **X**MLの略
- サーバーとの非同期な通信
 - JavaScript の XmlHttpRequest を使用
- 既存の技術の組み合わせ
 - JavaScript / DHTML / CSS
- インテリジェントなユーザーインターフェイス
 - ブラウザ上でユーザーとシームレスにやり取り
- Technology ではなく Technic
 - Ajax という具体的な何かがあるわけではない



JavaScript Library



- Prototype.js 1.4.0
 - ほぼスタンダードとなっている、JavaScript のライブラリ
 - クラス指向のように JavaScript を記述できる
- Rico 1.1.2
 - Prototype.js を使用した Ajax アプリケーションの構築用ライブラリ
 - ユーザーインターフェイスへの Drag & Drop やエフェクト機能などの提供

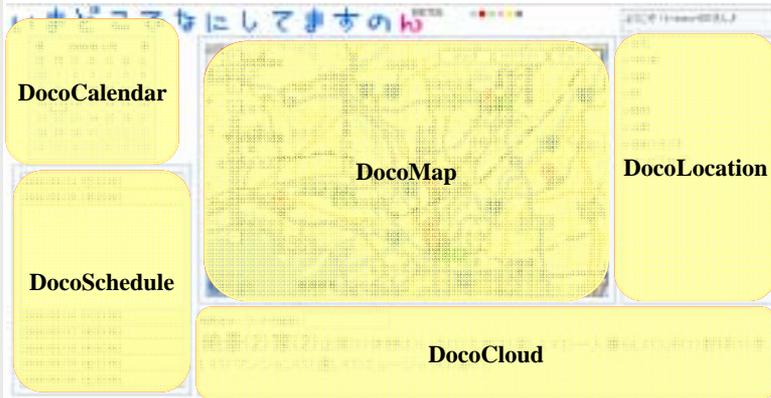


コンポーネント分割

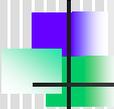


- コンポーネント (DocoComponent) としてラップ
 - Calendar , Schedule , Map , Location , Cloud

XML Consortium



© XML Consortium

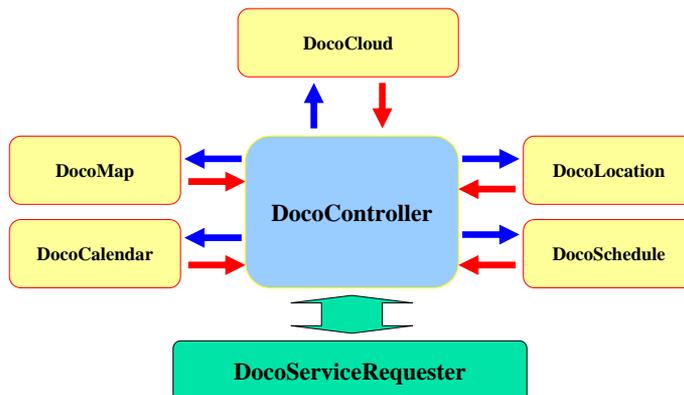


コントローラー



- 中央にコントローラーである DocoController を配置し各コンポーネントを生成
- DocoController はイベントを使用してコンポーネントと通信

XML Consortium

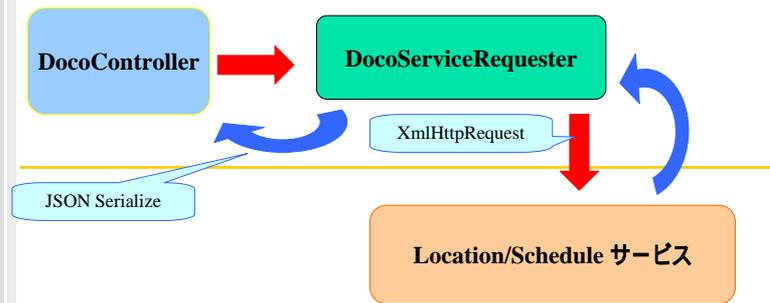


© XML Consortium

サービス接続



- データの取得は DocoServiceRequester を介して Web サービスに直接つなぐ
- Soap Response を JSON 形式にシリアライズしてクライアントにコールバック

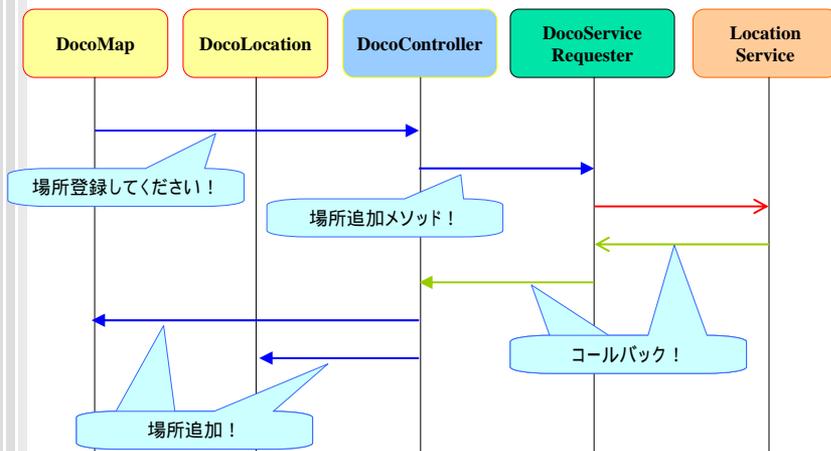


© XML Consortium

流れ



例: 場所登録



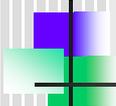
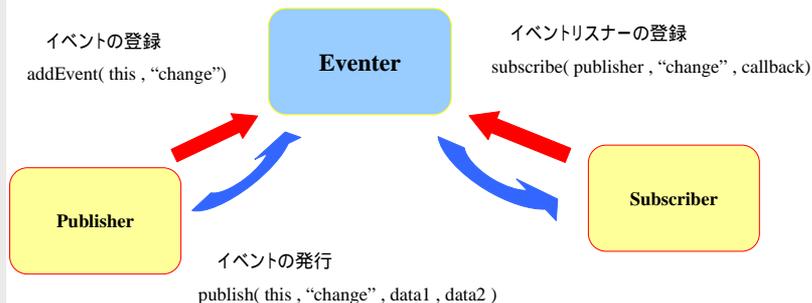
© XML Consortium



イベント



- イベントを管理する Eventer
 - Publish / Subscribeパターン
 - インスタンスとキーで管理するのでどんなオブジェクトでも登録可能



今後の展開



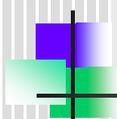
- Doco シリーズを展開
 - 場所情報などの画像アップ機能 (DocoImage)
 - 更新情報のRSS配信 (DocoRSS)
- 拡充
 - データの更新・削除
 - UCD の向上

もっと使えるようになります！
まだまだベータ版



まとめ

コベルコシステム株式会社 幸田 健



明らかになったこと



- 技術的な側面
 - Webアプリケーションでの、JavaScriptの必要性が増大している
 - 各種AJAXをサポートするライブラリが充実してきている
 - オープンソースが充実
 - ライブラリ、コントロール(マップやカレンダー)など、
 - 複数のライブラリを同時に使う場合、個々のコントロール(UIライブラリ)をどうつなげるかが課題
 - イベント機構を作りこみラップするなど
 - JavaScript言語における特有の動作
 - 手続き的にもオブジェクト指向的にも扱うこともできる
 - Webサービスを呼び出すには手間がかかる
 - SOAPなどNameSpaceを含むXMLを扱う場合
 - マルチブラウザへの対応は十分なテストを
 - ライブラリ(Prototype.jsなど)で差異を吸収
 - デバッグ環境の構築は必須
 - Firefoxの機能拡張(JavaScript Debugger)など

メンバーの所感



- Web2.0的アプリケーションの使う「楽しさ」と創る「楽しさ」
 - Web2.0的なサンプルを試してみるのは楽しかった
 - Web2.0的サービスと言われるものがどんなものなのか、実装を通して体感できた
 - JavaScriptはやっぱりいけている。楽しい。
- 技術の習得、ノウハウの蓄積
 - AJAX(DHTML/JavaScript/CSS)の技術が身についた
 - とにかく、本やWebで情報収集して実装に取り組んだ
- 分散開発環境の利用技術の蓄積
 - 分散環境での開発は、比較的スムーズに行えた
 - trac等のツールを実地に使うことができて、ノウハウも蓄積できた
- 新技術への対応
 - 認証サービスについては、インターネット上で認証情報をやり取りするための技術がいろいろ出てきているようなので、今後取り入れていきたい

まとめ



- Web2.0的アプリケーションの開発を実践
 - 分散開発環境の基盤を構築
 - 「使える」アプリケーションを意識
- 今後(今まで作ってきた環境を基盤として)
 - もっと使いやすく改善
 - サービスの機能追加(Geocoding, 天気, 仲間一覧...)
 - 新技術への対応