RADServer活用ショーケース IoTを活用してデータを集積

第33回 エンバカデロ・デベロッパーキャンプ

エンバカデロ・テクノロジーズ
セールス コンサルタント
井之上 和弘

embarcadero
DEVELOPER CAMP

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

はじめに

モバイルデバイスからのアクセスやIoTデバイスからの収集では、中間サーバーとなるWeb APIが重要な役割を果たします。Delphi/C++Builder でこうした機能を構築できるRAD Serverは、これまで作成したコードを流用してWeb API を開発可能です。このセッションでは BeaconFence で取得した位置測位情報を RAD Server 経由でバックエンドのデータベースに格納するという事例をベースにその構築方法を紹介します。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

アジェンダ

- 事例の紹介
  - 西都原考古博物館の事例紹介（第32回デベロッパーキャンプに未参加の方向けのフォローアップ）
- 構築方法
  - 基本構成の紹介
  - RAD Server の利用で考慮すべき内容

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

▪事例の紹介

- 西都原考古博物館の館内ナビアプリ向けにBeaconFenceを導入した。（第32回デベロッパーキャンプでの事例紹介から抜粋）
- 来館者の動きや滞留状況の分析を行うために、BeaconFenceで測位した位置情報の収集を行うように改修を実施。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

西都原考古博物館について

- 西都原古墳群
  - ３世紀末から７世紀にかけて築造
  - 陵墓参考地の男狭穂塚・女狭穂塚を加えた319基
    - 前方後円墳31基、方墳２基、円墳286基
- 博物館はこの古墳群の一角に位置
  - 考古学専門のフィールドミュージアムとして、2004年4月に開館
  - 南九州の歴史を学ぶことが可能

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

展示室

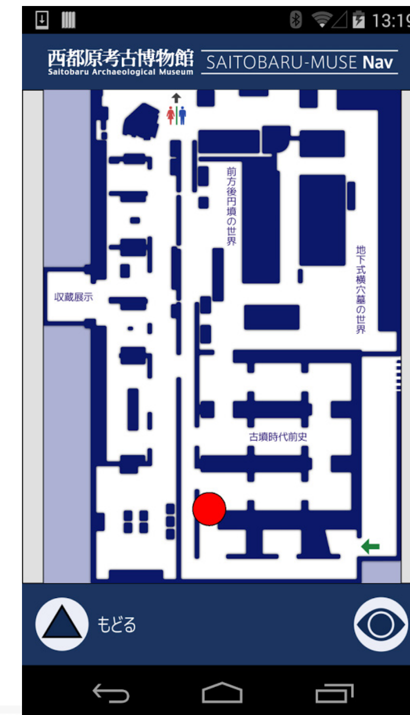
- 縄文時代から古墳時代の歴史を展示
- 地下式横穴墓の原寸大模型
- 大型の地形模型が西都原古墳群の立地と古墳の分布状況の理解をうながす

saito-muse.pref.miyazaki.jp

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

西都原考古博ナビ

- iOSおよびAndroidで利用可能
- RAD StudioおよびBeaconFenceによって開発
- 来場者に館内のナビゲーションと４言語による情報提供
  - 現在位置をマップに表示
  - 好みの言語で展示情報を表示
    - BeaconFenceのゾーン機能により、現在位置に対して適切な情報を表示
    - QRコードによる情報の表示にも対応
- スタッフの労力をかけることなく、海外からの来場者に対して、多言語による情報提供が可能に



本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

■位置情報を収集する

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

収集した情報の利用用途

- 来館者全体を対象としたヒートマップ分析
  - ナビアプリの利用状況を把握する
  - 展示構成や順路の改善の参考に用いる

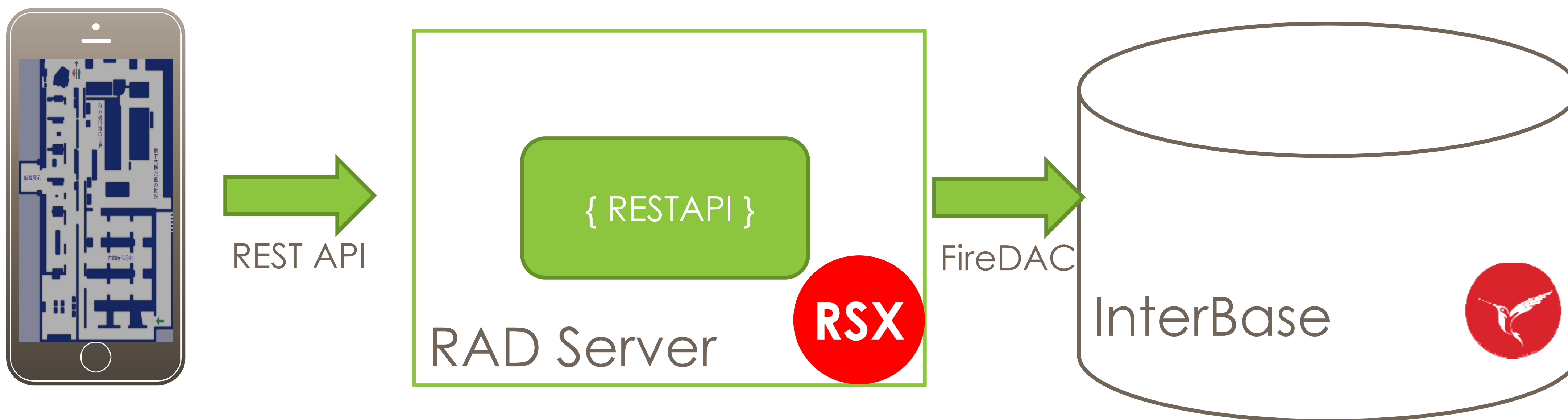
本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

▪構築方法

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

基本構成

- モバイルクライアントの位置情報は **REST API で RAD Server に送信**する。
- バックエンドデータベースには **FireDAC で接続**する。



本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

RAD Server で出来ること

- 「EMS管理用API」の利用 （EMS = エンタープライズモビリティサービス）
  - **ユーザー管理、認証、グループ管理**
  - プッシュ通知
  - 登録デバイス管理
  - アナリティクス機能（APIの利用状況になどの分析）

- 「カスタムEMSリソース」の利用
  - **「 EMS管理用API 」以外のRESTFulなWebAPI機能をDelphi/C++Builderで実装できる**
  - カスタムEMSリソースの実装では、クライアント側アプリで収集するデータの型やクラスの実装を流用することで開発の効率化が図れる。

- BeaconFence
  - **BeaconFenceを適用エリアの面積やビーコンの個数によらず利用できる**

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

RAD Server の利用で考慮すべき内容

- 収集するデータの選定
- RESTエンドポイントの設計
- 認証処理
- 通信方法の検討

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

▪収集するデータの選定

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

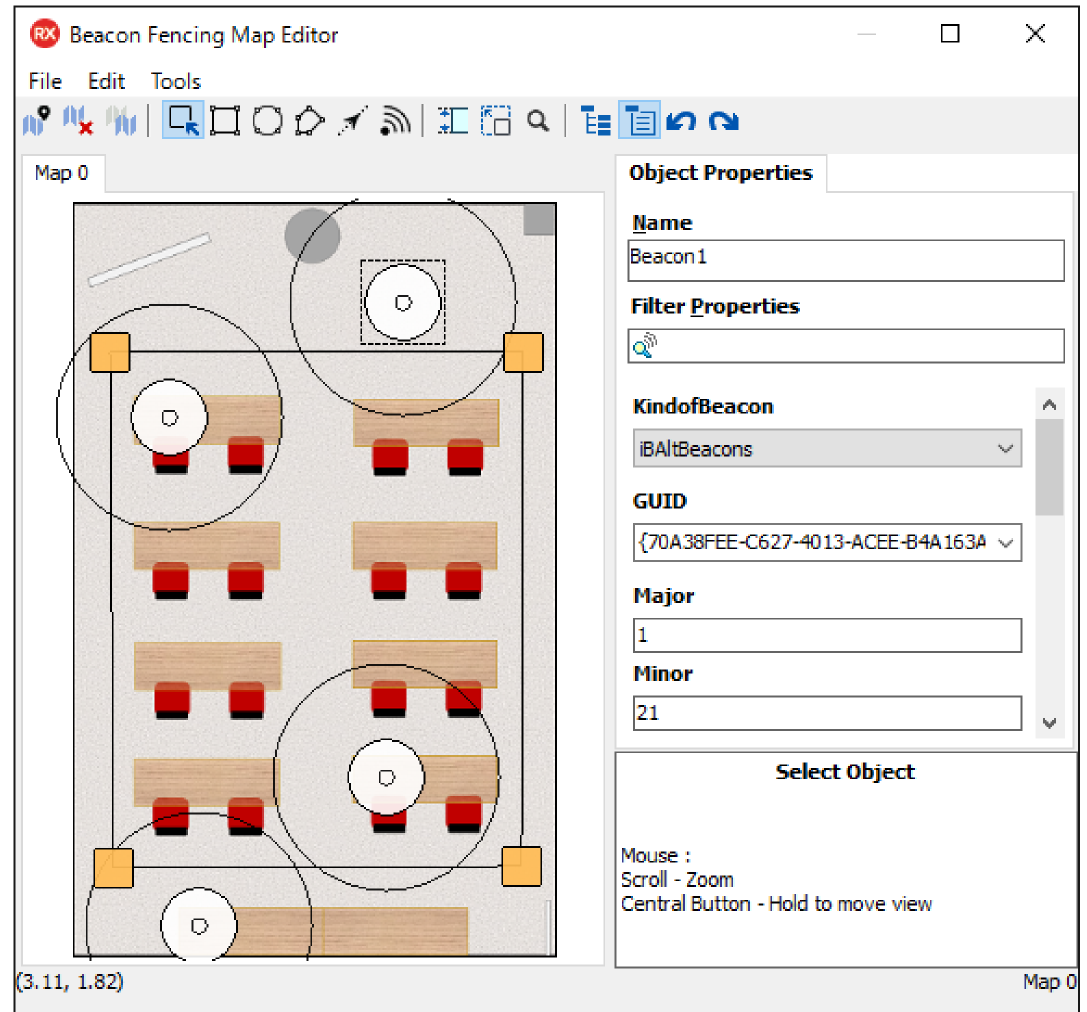
補足: Beaconfenceで測位アプリを作る。

- フロアマップを定義し、ビーコンを配置する。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

補足: BEACONFENCEで測位アプリを作る。

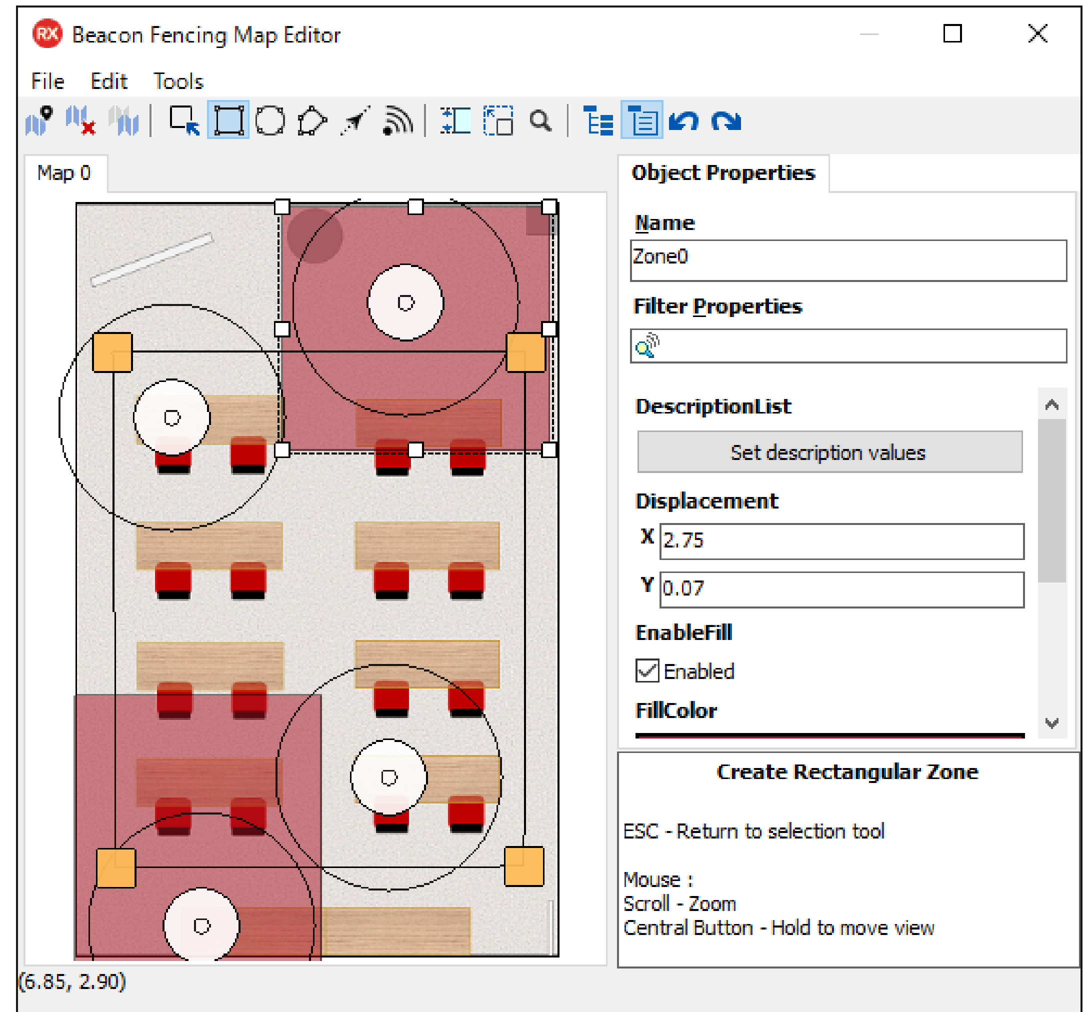
- **測位精度を高めるために、人の動線を「パス」で定義する。**
  - 「パス」は、カーナビにおける道路に相当する。
  - ビーコンの測距、測位精度はさほど高くないが、パスを定義することで測位位置の補正が行える。



© 2016 Embarcadero Technologies, Inc. All rights reserved. Proprietary and confidential.

補足: BEACONFENCEで測位アプリを作る。

- **特定のエリアの出入りに伴って特別な処理を自動で行いたい場合は、「ゾーン」を定義する。**
  - ゾーンを定義すると、ゾーンへの出入りに伴う処理を実行できるようになる。
  - 例えば「ゾーン固有のメッセージや画像を表示する」「そのゾーンの展示物の説明音声や動画の再生処理を起動する」など。



© 2016 Embarcadero Technologies, Inc. All rights reserved. Proprietary and confidential.

収集するデータの選定 - BeaconFenceで発生するイベントの例

イベント	意味	取得可能なデータ
OnPositionEstimated	位置測位を行った	**計算上の位置、パスへの補正後の位置**
OnZoneEnter OnZoneExit	ゾーンへの出入りを検出した	**出入りしたゾーンの情報**
OnBeforeMapChange	表示に使用するマップの切り替えを行う直前のイベント	**切り替え後のMapID**、切り替え前のMapID
OnBeraconEnter OnBeaconExit	ビーコンのエリアに入った ビーコンのエリアから出た	ビーコンの識別情報、近接度、電波強度、距離
OnBeaconProximity	ビーコンの近接度が変化した	ビーコンの識別情報、近接度、電波強度、距離 現在のゾーンの情報
OnBeaconEddyTLM OnBeaconEddyURL	EddyStoneビーコンのTLM情報を検出した。 EddyStoneビーコンのURL情報を検出した。	テレメトリー情報（稼働時間、バッテリ残量、温度） URL情報

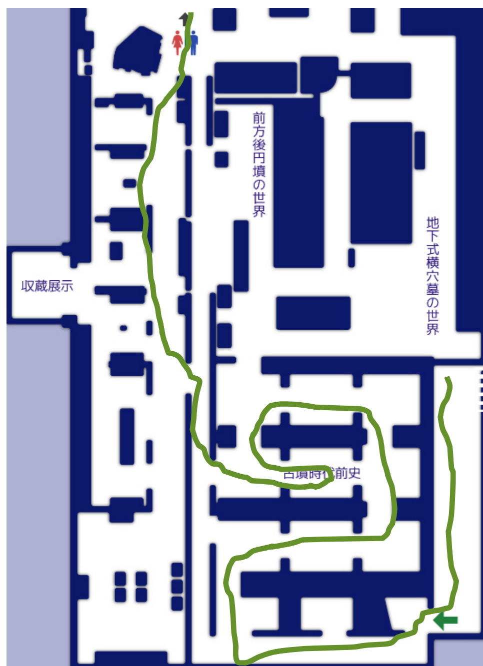
※OnBeaconEddyTLM, OnBeaconEddyURLは 10.1 Berlin 以降での対応。10 Seattle では非対応。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

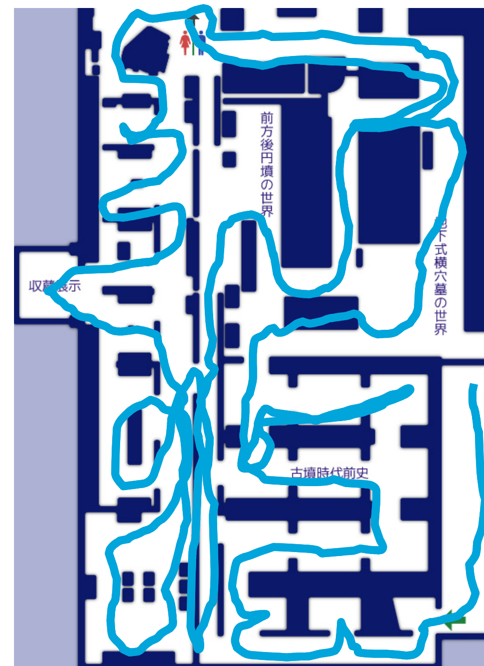
収集するデータの選定 - 端末を区別するための情報

- 端末ごとの行動履歴はトレースできるべき。ただし個人情報との紐づけは必須ではない。
- **UUIDのように重複のおそれが少ないIDを添えて行動履歴を記録する。**

端末Aの移動経路



端末Bの移動経路



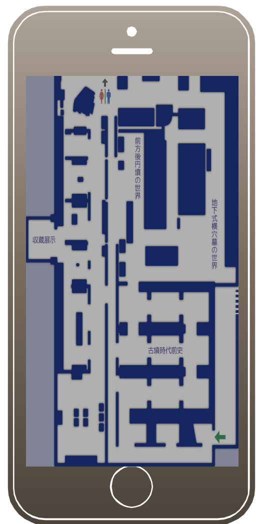
両者の区別が必要

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

収集するデータの選定 - 日付・時刻

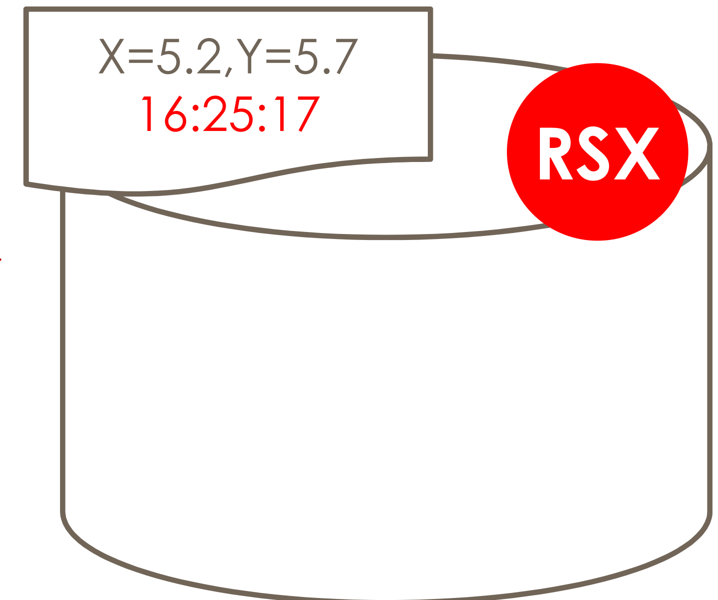
- **日付時刻は端末側の時計を用いる。**
- サーバ側でタイムスタンプを振るのは好ましくない。通信遅延等によって不正確な値となる。

Time=16:25:07



X=5.2,Y=5.7
(w/o timestamp)

X=5.2, Y=5.7
16:25:07



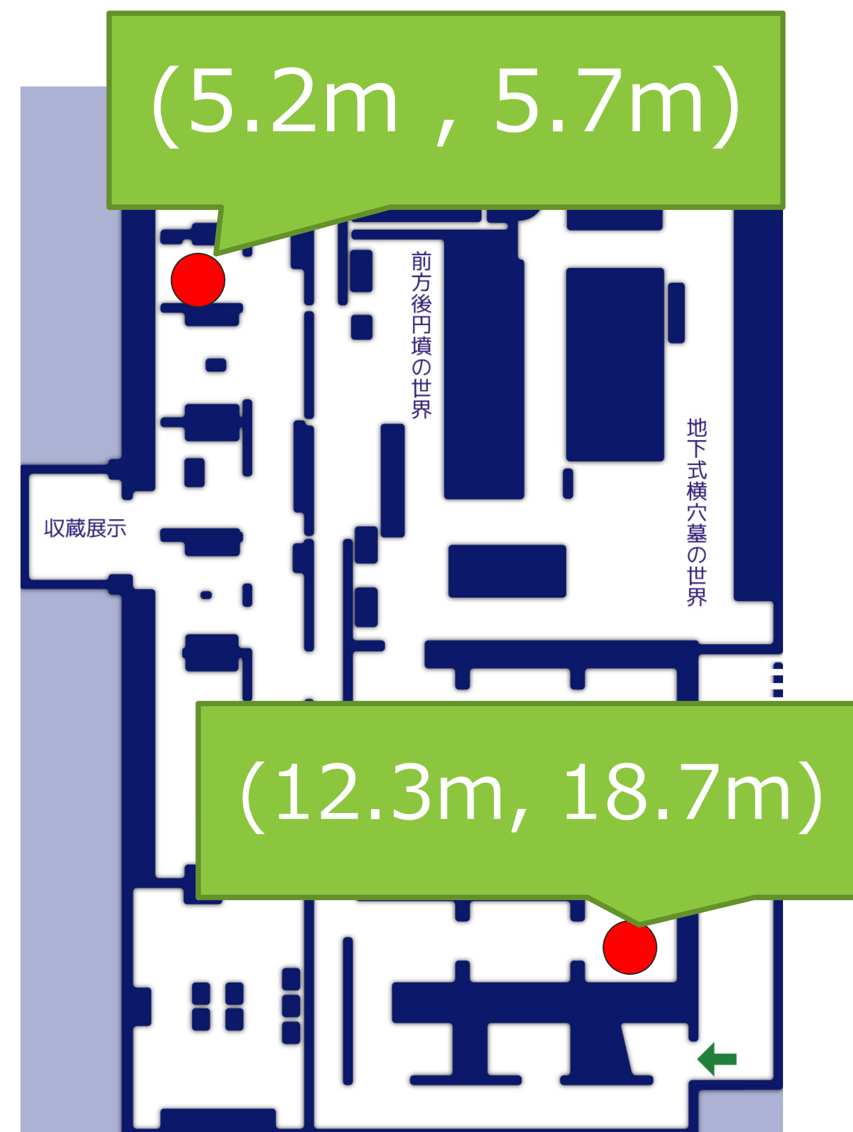
本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

収集するデータの選定 - 位置情報

- マップ上での位置情報は測位完了のイベントで収集する。

```
// Delphi
procedure TForm1.BeaconMapFencing1PositionEstimated(
  AEstimatedPoint, APointToPath: TPointF);
begin
// 計算上の位置 = AEstimatedPoint
// 補正後の位置 = APointToPath
end;
```

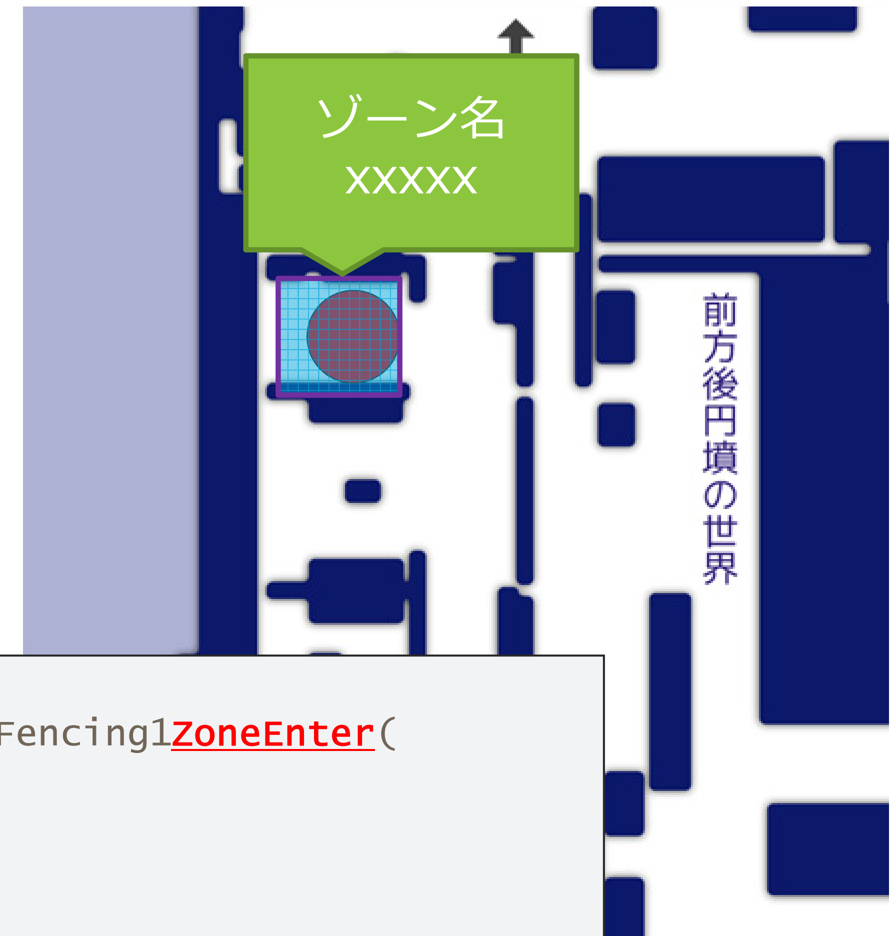
```
// C++
void __fastcall TForm1::BeaconMapFencing1PositionEstimated(
  TPointF &AEstimatedPoint,
  TPointF &APointToPath)
{
// 計算上の位置 = AEstimatedPoint
// 補正後の位置 = APointToPath
}
```



本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

収集するデータの選定 - ゾーン情報

- マップ上に設定されたゾーン内にいた場合は、そのゾーン情報も収集する。
- 必要に応じてゾーンから出た場合にも処理を行う。




```
// Delphi
procedure TForm1.BeaconMapFencing1ZoneEnter(
  const Sender: TObject;
  const AZone: IDesignZone);
begin
// ゾーン名 = AZone.Name
end;
```

```
// C++
void __fastcall TForm1::BeaconMapFencing1ZoneEnter(
  TObject * const Sender,
  IDesignZone * const AZone)
{
// ゾーン名 = Azone->Name
}
```

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

収集するデータの選定 - MapIndex

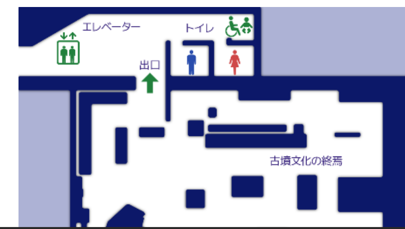
- BeaconFenceでは館内マップを複数のマップに分割して作成できる。
- 「マップ分割時はMapIndex + 位置情報」の組み合わせが意味を持つので、MapIndexも忘れずに収集する。

MAP 1

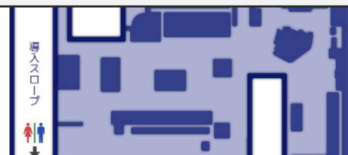
MAP 2

MAP 3

MAP 4




```
// Delphi
procedure TForm1.BeaconMapFencing1BeforeMapChange(
  const Sender: TObject;
  CurrentMapIndex, NewMapIndex: Integer; var Allow: Boolean);
begin
// NewMapIndex = 表示切替後のMAP
end;
```



```
// C++
void __fastcall TForm1::BeforeMapChange(
  TObject * const Sender,
  int CurrentMapIndex, int NewMapIndex, bool &Allow)
{
// NewMapIndex = 表示切替後のMAP
}
```

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

使用するデータと収集元イベントの一覧

データの種類	発生元イベント（または生成取得方法）
端末を区別するための情報	UUIDを生成して用いる
日付時刻情報	端末側の時計を用いる
現在の位置情報	OnPositionEstimated
現在位置を含んでいるゾーン情報や、ゾーンから外れた場合の検出	OnZoneEnter, OnZoneExit
表示に使用したマップ情報	OnBeforeMapChange の NewMapIndex

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

▪RESTエンドポイントの設計

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

エンドポイントとリソース

- RAD Server のエンドポイントは以下のような形式をとる。(本番運用時)
  - Microsoft IIS
    - http(s)://hostname/**emsserver/emsserver.dll**/**{**カスタム**EMS**リソース名**}**/**{*resource_suffix*}**
  - Apache httpd
    - http(s)://hostname/**emsserver**/**{**カスタム**EMS**リソース名**}**/**{*resource_suffix*}**

- カスタムEMSリソース名は、EMS向けのプロジェクトに設定する。

- ***resource_suffix*** はそれぞれのエンドポイントに対応する関数やプロシジャーごとに設定する。固定文字列だけではなく、可変パラメータを取り扱うことも可能。

- 赤文字の箇所をURLから隠したい場合は、IISやApacheの前段に配置するロードバランサーやリバースプロキシでマッピングを調整する等の対処を行う。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

エンドポイントとリソース

- 位置情報の送受信に用いる resource suffix の例。

Operation	resource_suffix	Function
POST	v1/location	測位の履歴を記録する
GET	v1/location/{id}	特定のUUIDに関する測位の履歴を取得する
GET	v1/idlist	UUIDのリストを取得する

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

RAD Serverとの通信に用いるコンポーネントを検討する

- TEMSFireDACClient
  - FireDAC のデータ形式を RAD Server 経由で取り扱える。
  - RAD Server と送受信するデータはJSONだが、コンポーネントの内部でJSONを取り扱うため、実装時にJSONを扱う必要はない。
  - TFDMemTable で結果を扱える。通常のデータベースアプリケーションと同様の実装が可能。
  - 但し一般的なRESTful APIに比べて送受信するデータ量や構造が冗長となる。

- TBackendEndpoint
  - RAD Server のエンドポイントと通信するための汎用コンポーネント。
  - 送受信するデータは JSON を自前でハンドリングせねばならない。
  - データの内容は一般的な RESTful API と同様のシンプルなものとなる。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

西都原考古博ナビアプリでは、 TBackendEndpointを用いる

- 来館者のモバイル回線に流れるデータ量は必要最小限にしたい。
  - ナビアプリは一般来館者の端末とモバイル回線を使っている

- 位置情報のデータ量が、そもそも小さい。
  - TEMSFireDACClient のように高機能なコンポーネントを用いてデータ量が増えることは好ましくない。
  - ただし業務用機器のように回線コストをエンドユーザが負担しない場合は、データ形式の複雑さによらず TEMSFireDACClient の利用を検討すべき。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

■認証処理

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

ユーザとグループの設計

- 個別のインストールベースごとにユーザアカウントを発行するか否か？
  - 今回の実装では**個別のユーザアカウントを発行しない。**
  - 各個人がログインしての利用形態は想定しない。
  - ただし**個別のデバイスの識別は必要。**

  - **アプリの初回起動時に識別用のUUIDを生成発行**して、位置情報履歴とセットで記録する。

  - **認証無しでのデータ送信は受け付けない。**アプリにプリセットした認証情報を用いてサーバとの通信を行う。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

ユーザとグループの設計

- グループは「**管理者**」と「**来館者**」の２つを定義する。

- 「**管理者**」グループには「**データ閲覧ユーザ**」が所属する。
  - 「データ閲覧ユーザ」では蓄積されたデータの取得を許可する。

- 「**来館者**」グループには「**位置データ送信用アカウント**」が所属する。
  - このアカウントでのRAD Serverアクセスは位置情報履歴の送信だけを許可する。
  - その他の RAD Server アクセスはすべて拒否。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

RAD Server の認証処理は設定ファイルで行う

- 認証処理のためにコードを書く必要がなく、**設定ファイル EMSServer.ini だけでも処理できる**。
- **初期設定では認証設定が無い**。すべての標準リソースやカスタムリソースを利用できる。
- ログイン認証が成功したユーザだけを対象にリソースの利用を許可するには、個々のリソースに対する認証設定を EMSServer.ini に記述する。
- EMSServer.iniでの制限がなされていないリソースについて、カスタムリソース側の実装でアクセスコントロールを行うことも一応は可能。

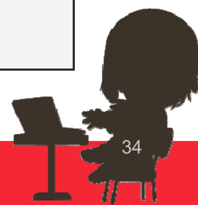
本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

認証処理の記述例 – EMSServer.ini

```
[Server.Authorization]
; Version, Users, Groups, Installations, Push, Edgemodule は
; 一切のアクセスを禁止する。
Version={"public": false}
Users={"public": false}
Groups={"public": false}
Installations={"public": false}
Push={"public": false}
Edgemodules={"public": false}

; Users.LoginUser は認証処理に使用するため、未ログイン状態のアクセスを受け付ける。
Users.LoginUser={"public": true}

; カスタムEMSリソースは group = visitor に所属する認証済みユーザだけがアクセスできる。
カスタムEMSリソース名={"public": false, "groups": ["visitor"]}
```



本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

ログイン処理

- RAD Server ではクライアントアプリからのログイン処理は、**アプリ内で１回だけ実施**しておけばよい。

- 以後のサーバへの通信では、キャッシュされたログイン情報を用いるので、ログイン処理は省略できる。

- ただし通信エラーの発生時はログイン処理を念のために再度実施する。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

ログイン処理の実装例

- **ログイン処理は別スレッド**で行った。
  - BackendAuth.Loginではログインに伴うタイムアウト時間を変更できない。
  - 通信遅延やサーバ障害が発生すると処理のブロッキングの恐れがあるので別スレッドでの処理が必要。
  - 無名スレッドを用いればスレッド処理をシンプルに実装できる。

```
// ログイン処理をDelphiの無名スレッドで実行する例。
TThread.CreateAnonymousThread(
  procedure()
  begin

  if (not FLogin)  then
    begin
      try
        BackendAuth.Login;
        FLogin := True;
      except
        FLogin := False;
      end;
    end;
  end
).Start;
```

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

▪通信方法の検討

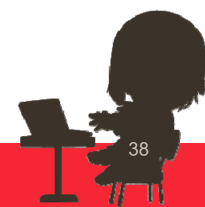
本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

HTTPS の利用は必須

- iOSアプリでは通信に HTTPS の利用が必須。
  - Appleの方針により、2017年以降はHTTPSが原則必須。

- ただしEV証明書は不要。アプリの内部通信でEV証明書を検証する意味は少ない。
  - RAD ServerへのREST APIアクセスではブラウザのように任意のURLを扱うことをしないため、EV証明書の情報をアプリ画面内に常時表示する意味がない。

- **証明書は案件の内容や要件に合わせて選定する。**
  - Amazon Web Service なら AWS Certificate Manager の無償証明書。
  - オンプレミスなら Let's Encrypt の無償DV証明書が利用可能かを検討する。
  - 上記が利用できないなら、従来通りに商用のSSL/TLS証明書をどうぞ。



本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

測位した情報の都度送信は極力避ける

- クライアント側のデータ通信量に対するインパクトが大きい
  - 測位１回分のデータ量は、HTTPヘッダよりも小さいので都度送信は効率が悪い。

- 測位結果の都度送信はサーバ側の負担も増す
  - 位置測位は１秒あたり２回前後程度発生する。
  - 仮に40端末が同時に用いられると、１秒間80件、１分間では4800件の処理が発生する。
  - 個々の端末からの送信を１分毎にまとめれば、サーバ側の処理は１分あたり40件（=40端末の分）で済む。

- <u>**リアルタイムの分析が不要ならば、測位した情報を都度送信することは避けて、一定時間分をまとめて送信する。**</u>

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

まとめ

- 収集するデータの選定
  - 必要なデータが一発で揃うわけではないので、個々のイベントで発生した値を適切に保持して使う。
- RESTエンドポイントの設計
  - カスタムリソース名、resource_suffix を要件に合わせて適切に設計する。
  - 通信に用いるコンポーネントは、使用する状況に合わせて選定する。
- 認証処理
  - 初期設定はすべてのリソースに対してフルアクセスなので、設定ファイルを調整して適切に設定する。
- 通信方法の検討
  - データの都度送信は極力さける。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

THANKS!

www.embarcadero.com/jp

第33回 エンバカデロ・デベロッパーキャンプ

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。