






# ITアーキテクチャ

<div>ITアーキテクチャ</div> <div>サービス開発のエンジニアリングガイド</div> <div> 協業開発チーム 2018/10/01  6</div>	<div>ITアーキテクチャ</div> <div>コーディング規約</div> <div> Nablarchチーム 2022/11/02  22</div>
<div>ITアーキテクチャ</div> <div>Javaアプリケーションフレームワークと設計・開発をサポートするコンテンツの活用</div> <div> Nablarchチーム 2022/11/02  14</div>	<div>ITアーキテクチャ</div> <div>SPA + REST API構成のサービス開発リファレンス</div> <div> 協業開発チーム 2020/09/30  15</div>

Update

## 目次

- [本カテゴリーの位置付け](#)
  - 隣接カテゴリー
    - Nablarch
- [コンテンツ紹介](#)
  - [サービス開発のエンジニアリングガイド](#)
  - [コーディング規約](#)
  - [Javaアプリケーションフレームワークと設計・開発をサポートするコンテンツの活用](#)
  - [SPA + REST API構成のサービス開発リファレンス](#)

## 本カテゴリーの位置付け

本カテゴリーでは、特定のプロダクト等に依存しないITアーキテクチャ全般のトピックを扱います。

隣接カテゴリーとの関連を下図に示します。

Cookie利用について



本カテゴリーの位置付けと隣接カテゴリーとの関連

## コンテンツ紹介

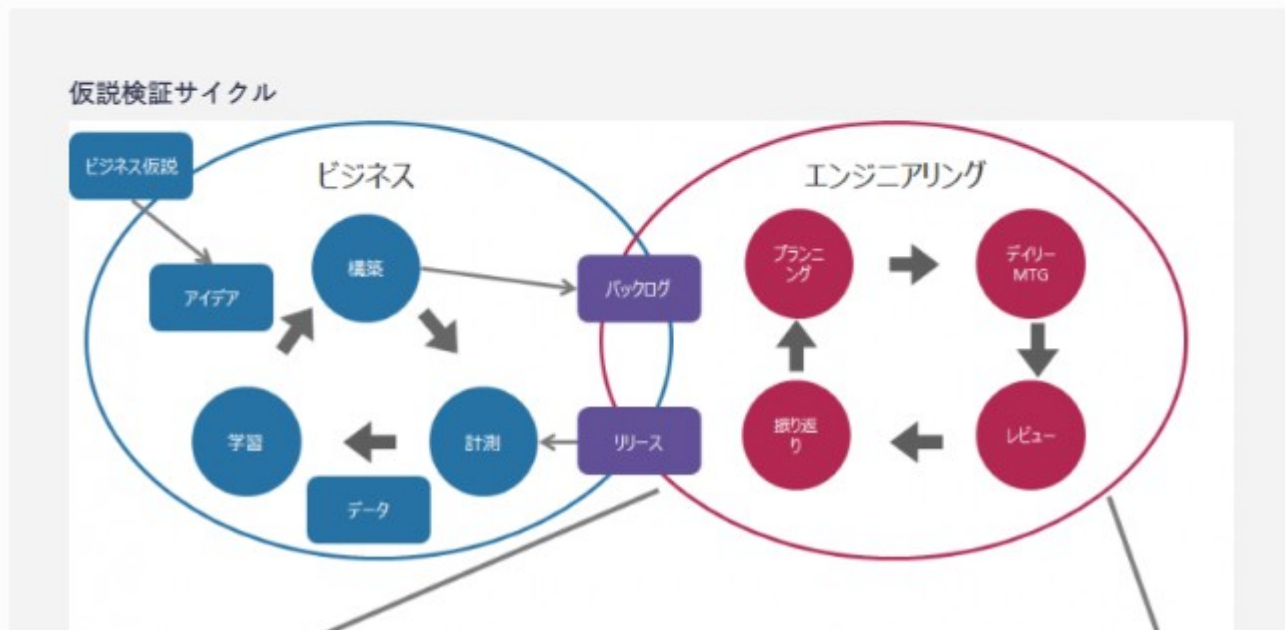
## サービス開発のエンジニアリングガイド

[サービス開発のエンジニアリングガイド](#) は、サービス開発に携わるエンジニア向けに、サービス開発の考え方、アーキテクチャ設計、開発プロセス設計などを概説しています。参考リンクから更に詳細を学ぶこともできます。

## サービス開発の考え方

サービス型ビジネスモデルにおいては、変化が速く不確実で予測困難な市場に対して、ビジネス面の仮説検証サイクルを繰り返し、市場の反応を見ながらサービスを改善して付加価値を高めていくことが重要になります。

このようなモデルでは、サービスの成功にはビジネス仮説の検証機会を増加させることが必要となりますが、そのためには、いかにしてビジネス仮説から市場へのローンチまでにかかる時間（リードタイム）を短縮するかが鍵になってきます。



サービス開発のエンジニアリングガイド

## コーディング規約

[コーディング規約](#) には、Javaのコーディング規約や静的解析ツール（Checkstyle、SpotBugs）の設定ファイル、コードフォーマッタ設定ファイルなどがあります。

5.11.秘匿情報はログ出力やシリアライズされないように注意してください

パスワードのような秘匿情報はログに含まれないようにマスクするなど、注意をしてください。

また、インスタンスをシリアライズする場合も、秘匿情報が含まれないように注意してください。なお、ここでは「シリアライズ」という言葉の意味として、JavaのシリアライズのみならずJSONやXMLなどのフォーマットに変換することも含めています。

例えばJavaのシリアライズの場合、シリアライズ対象外にしたいフィールドには `transient` キーワードを付与します。

```
public class LoginForm implements Serializable {  
  
    private String username;  
    private transient String password;  
  
    ...  
}
```

JSONやXMLへのシリアライズでも、通常はライブラリがシリアライズ対象外にする方法を用意してありますので、適切に対応してください。

5.12.Java標準ライブラリにあるレガシーなAPIは使用しないでください

Java標準ライブラリには過去のバージョンでは使われていましたが、今となってはレガシーであり使うべきではないAPIがあります。次に列挙するレガシーAPIは使わないようにしてください。

レガシーAPI	代替となる使っても良いAPI
<code>java.lang.StringBuffer</code>	<code>java.lang.StringBuilder</code>
<code>java.util.Dictionary</code>	<code>java.util.Map</code>
<code>java.util Enumeration</code>	<code>java.util.Iterator</code>

Javaスタイルガイド

Javaアプリケーションフレームワークと設計・開発をサポートするコンテンツの活用

[Javaアプリケーションフレームワークと設計・開発をサポートするコンテンツの活用](#)では、Javaアプリケーションの設計・開発に利用できる各種コンテンツの活用方法を概説しています。各種コンテンツが一覧化されており、どのようなコンテンツがあるのか把握できるようになっています。 [Nablarch](#) 向けコンテンツ、[Spring](#) 向けコンテンツ、フレームワーク非依存のコンテンツに分類されています。

提供しているコンテンツの一覧

コンテンツ		Nablarch	Spring	FW非依存
ランタイム	フレームワーク本体	<a href="#">Q</a>		
	テストングフレームワーク	<a href="#">Q</a>		
	拡張機能	<a href="#">Q</a>	<a href="#">Q</a>	
サンプル	実装例	<a href="#">Q</a>		
	設計書とソースコードの対応を示すサンプル	<a href="#">Q</a>	<a href="#">Q</a>	
開発ガイド	システム開発全体の進め方を記したガイド	<a href="#">Q</a>		
	フレームワークの解説書	<a href="#">Q</a>		
	プログラマー向けの実装ガイド	<a href="#">Q</a>	<a href="#">Q</a>	

Javaアプリケーションフレームワークと設計・開発をサポートするコンテンツの活用



[SPA + REST API構成のサービス開発リファレンス](#)では、SPA（シングルページアプリケーション）とREST APIで構成されるウェブアプリケーションを開発する際に参考になるコンテンツを用意しています。

以下の3つのコンテンツで構成されています。

- 方式設計ガイド
- サンプル（チャットアプリ）
- ハンズオンコンテンツ



## SPA + REST API構成の方式設計ガイド

### SPA + REST API構成の方式設計ガイド

このドキュメントについて

[バリデーション](#)

REST API設計

フロントエンド・バックエンド間通信

CORS（オリジン間リソース共有）

ファイルダウンロード

ファイルアップロード

XSS（クロスサイトスクリプティング）対策

CSRF（クロスサイトリクエストフォージェリ）

## バリデーション

フロントエンドのUIで入力された値のバリデーションは、フロントエンド・バックエンドの両方で行います。

バリデーションは次の2つのパターンがあります。

1. フロントエンドでのバリデーションに加え、バックエンドでも同様のバリデーションを行う
2. フロントエンドではバリデーションが行えず、バックエンドでのバリデーションのみを行う

### フロントエンドでのバリデーションに加え、バックエンドでも同様のバリデーションを行う

バリデーションをフロントエンドで行う理由は、バリデーションのためにHTTP通信する必要がなく、バリデーション完了後の正しい状態となっている値だけを送信できるためです。また、

SPA + REST API構成のサービス開発リファレンス — 方式設計ガイド

