UI5con

Extreme Freestyle

Pushing the limits of reusability and extensibility

Dimitar Fenerski, SproutSoft

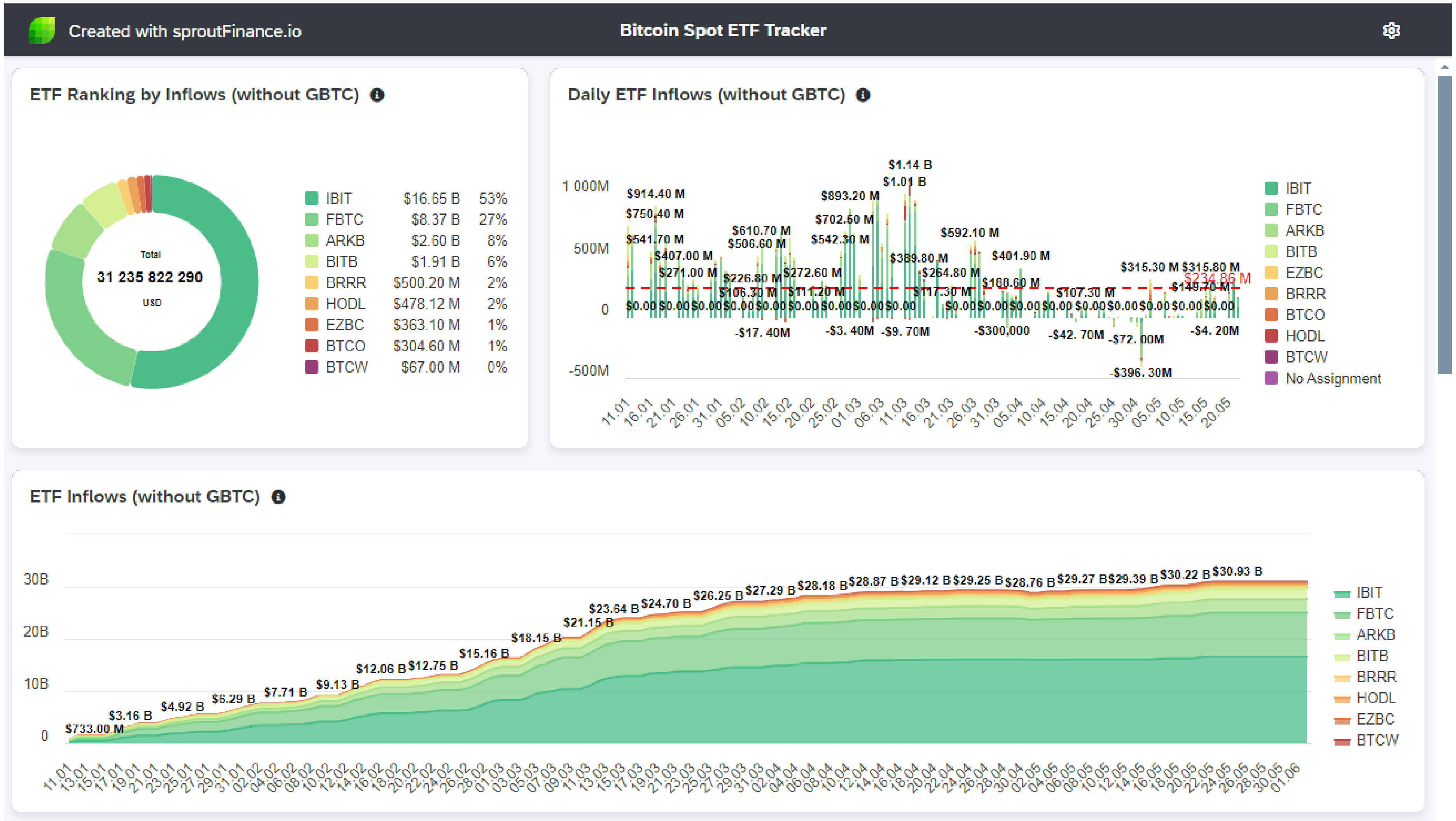
Building a SaaS with UI5

Project implications

- Not a typical scenario: building a product vs customer requested solution
    - 10000+ developer hours
    - 3 developers
    - Biggest non-SAP OpenUI5 project
- More “open” tech stack
- Rule-bending
    - SAP Fiori Guidelines
    - Diligent testing

UI5con

Is this Fiori?



Created with sproutFinance.io

Bitcoin Spot ETF Tracker

ETF Ranking by Inflows (without GBTC)

Total 31 235 822 290 USD

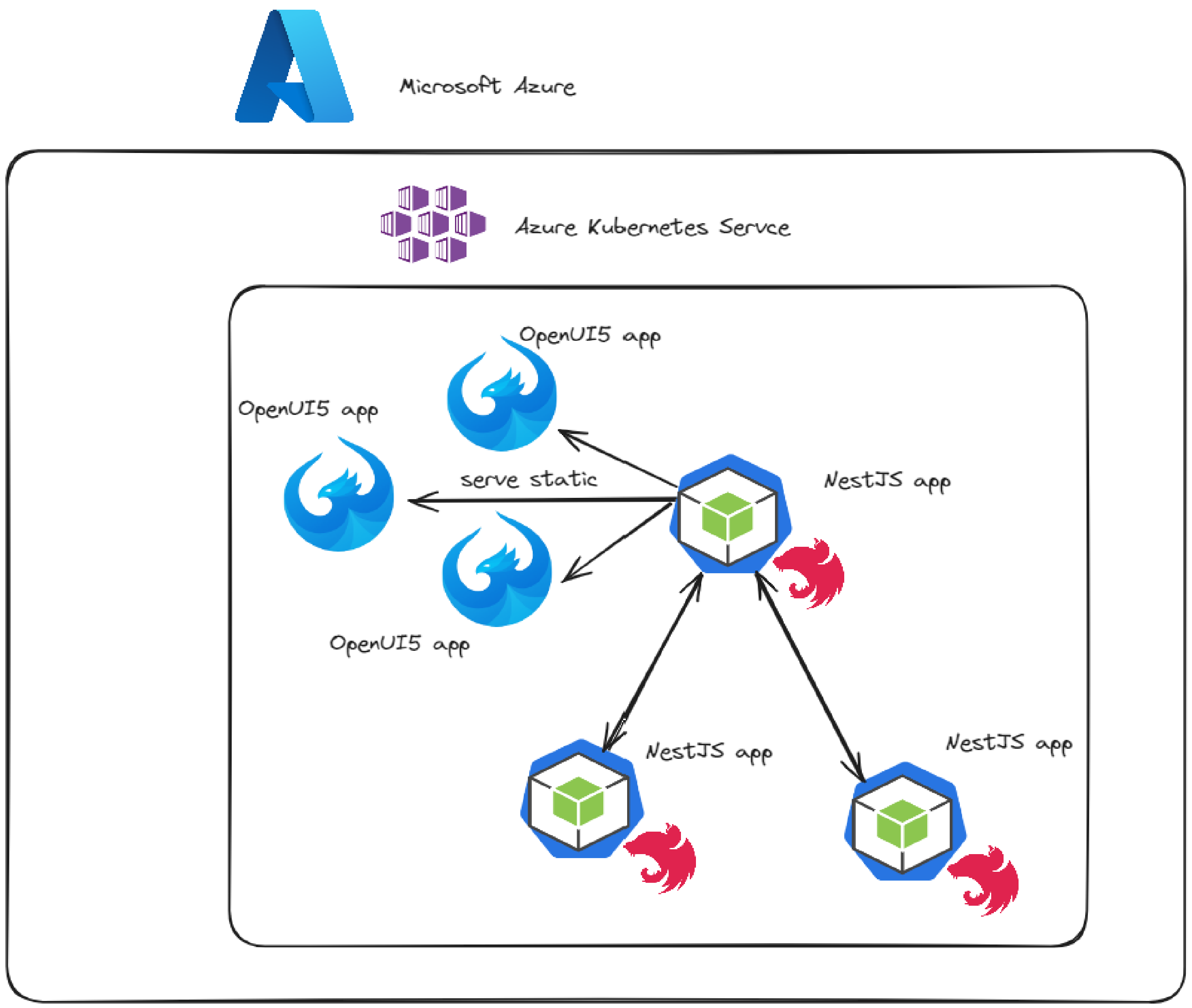
🟩	IBIT	$16.65 B	53%
🟩	FBTC	$8.37 B	27%
🟩	ARKB	$2.60 B	8%
🟨	BITB	$1.91 B	6%
🟨	BRRR	$500.20 M	2%
🟧	HODL	$478.12 M	2%
🟧	EZBC	$363.10 M	1%
🟥	BTCO	$304.60 M	1%
🟪	BTCW	$67.00 M	0%

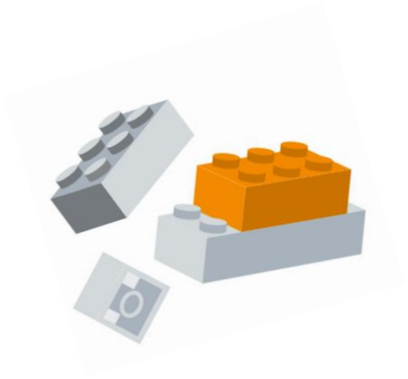
Daily ETF Inflows (without GBTC)

ETF Inflows (without GBTC)

UI5con

Architecture





UI5con

Building a SaaS with UI5

Frontend implications

- Only OpenUI5 – everything is freestyle
- No access to
  - SAP Flexibility
  - Fiori App Library
  - OData
  - Fiori Elements
  - …

How do we tackle this?

UI5con

Patterns overview

TS Everywhere

Library-first

Modular architecture

Type-safe state management

IoC using Dependency Injection

UI5con

TypeScript everywhere

Helps for: Extensibility

Introduction

- A startup's values:
  - Rapid prototyping
  - Concept evaluation
  - Fast iterations

- Facilitating work delegation

UI5con

Library-first approach

Helps for: Reusability

Library “standard” use cases

- Reusing a control definition
- Reusing utility classes

Library “extreme” use cases

- Reusing controllers
- Reusing business logic classes
- Reusing fragments

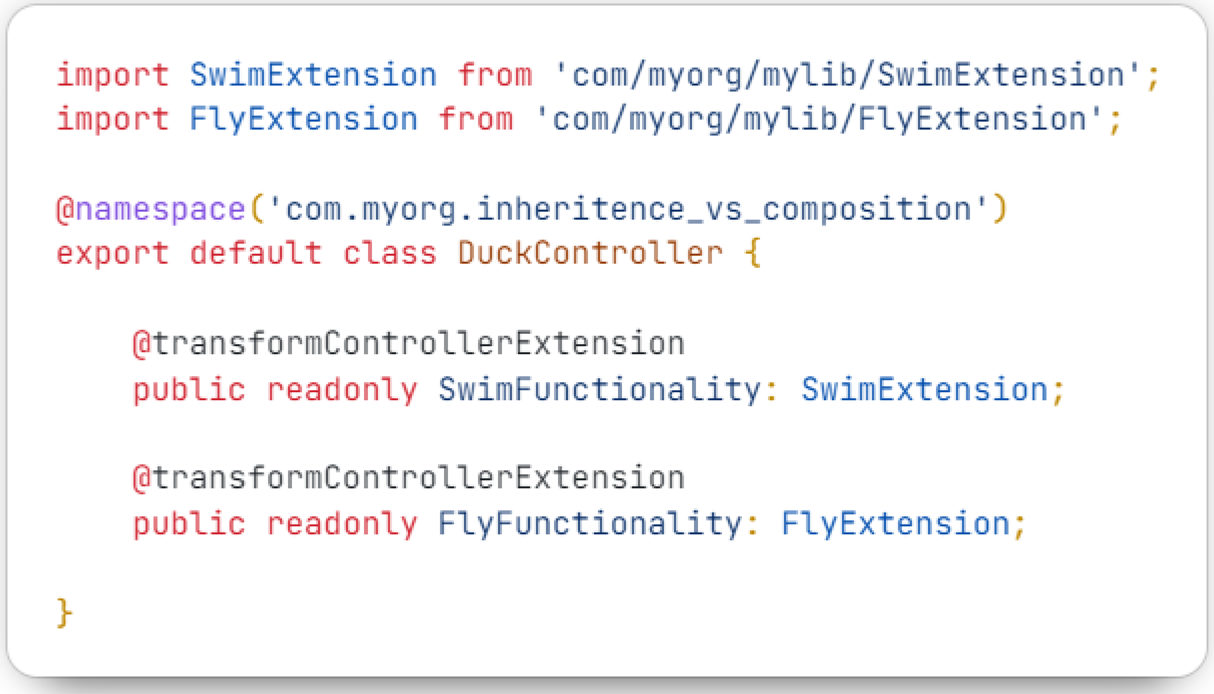
UI5con

Library-first approach

Helps for: Reusability

A hidden gem: `sap.ui.core.mvc.ControllerExtension`

- Inheritance vs Composition problem
- `ControllerExtensions` in libraries



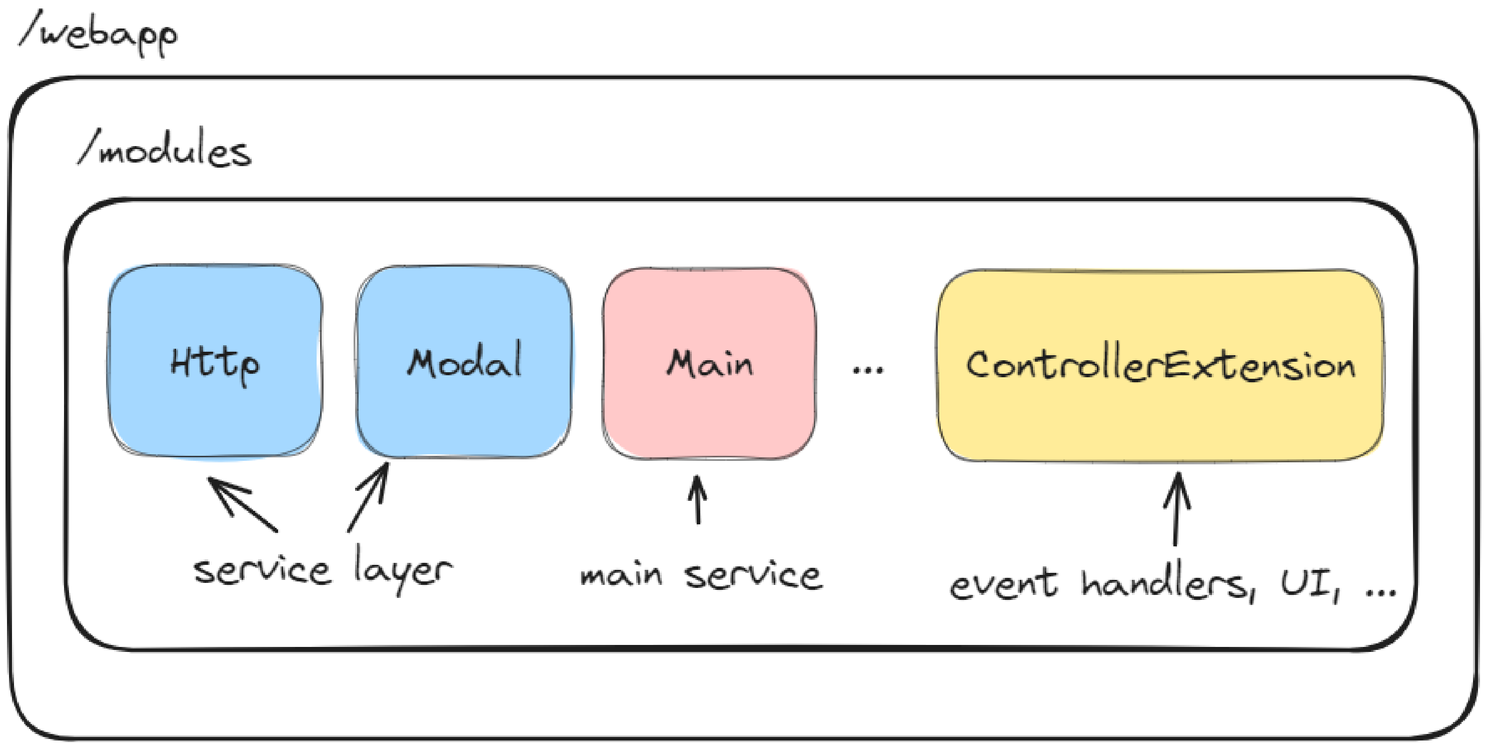
```
import SwimExtension from 'com/myorg/mylib/SwimExtension';
import FlyExtension from 'com/myorg/mylib/FlyExtension';

@namespace('com.myorg.inheritence_vs_composition')
export default class DuckController {

    @transformControllerExtension
    public readonly SwimFunctionality: SwimExtension;

    @transformControllerExtension
    public readonly FlyFunctionality: FlyExtension;

}
```

UI5con

Modular architecture

Helps for: Reusability, Extensibility

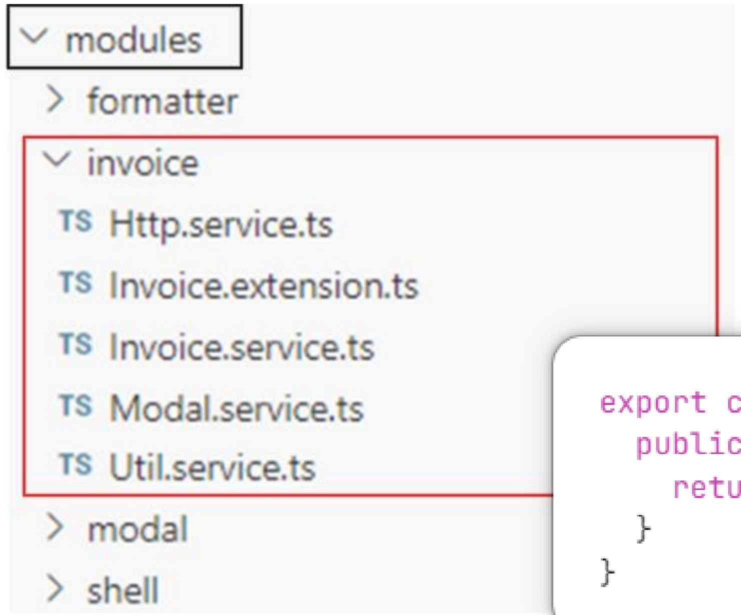
An overview of the pattern



Modular architecture

Helps for: Reusability, Extensibility

Example: Invoice module



```
export class UtilService {
  public getStats() {
    return Math.random()
  }
}
```

```
const API = 'https://api.example.com/v1/user';

export class HttpService {
  public get(userId: string) {
    return fetch(`${API}/${userId}`);
  }

  public patchEmail(userId: string, userEmail: string) {
    return fetch(`${API}/${userId}`, {
      method: 'PATCH',,
      body: JSON.stringify({
        userEmail
      })
    });
  }
}
```

UI5con

Strongly typed native state management

Helps for: Extensibility

A custom ui5 library

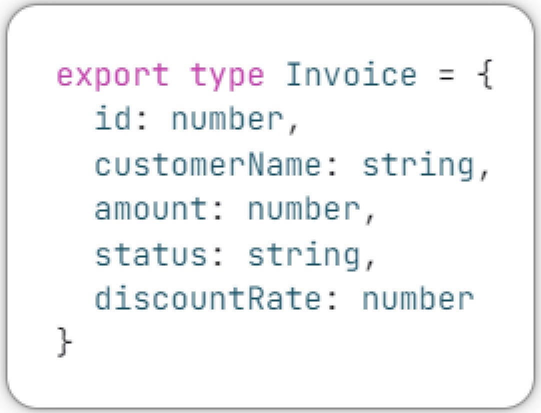
- Fully typed, **`DeepReadonly`** read access
- Streamlined write access
- References

```
import JSONModel from 'sap/ui/model/json/JSONModel';
import type { DeepPartial } from './types/DeepPartial';
import type { DeepReadonly } from './types/DeepReadonly';

export abstract class StateService<T extends {}> {
    protected readonly _model: JSONModel;

    protected get _data(): T {
        return <T>this._model.getData();
    }

    public get state(): DeepReadonly<T> {
        return <DeepReadonly<T>>this._data;
    }

    // ...

    protected set(data: DeepPartial<T>): void {
        this._model.setData(data, true);
    }
}
```

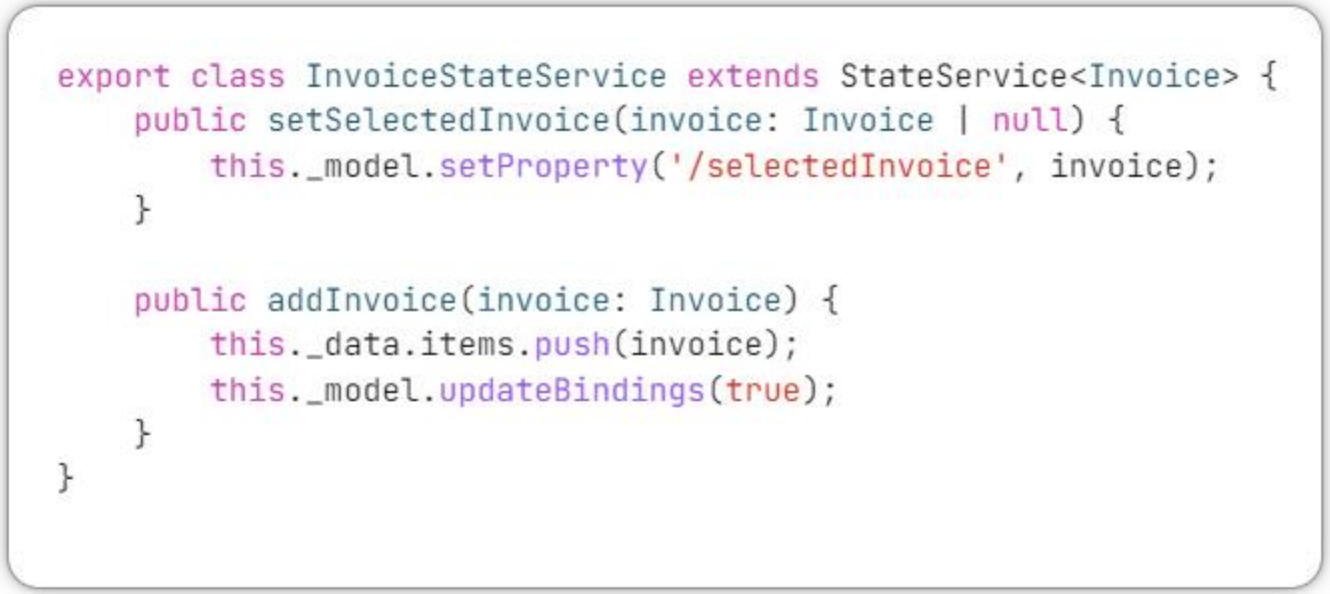
Strongly typed native state management

Helps for: Extensibility

Invoice state service



```typescript
export type Invoice = {
  id: number,
  customerName: string,
  amount: number,
  status: string,
  discountRate: number
}
```



```typescript
export class InvoiceStateService extends StateService<Invoice> {
    public setSelectedInvoice(invoice: Invoice | null) {
        this._model.setProperty('/selectedInvoice', invoice);
    }

    public addInvoice(invoice: Invoice) {
        this._data.items.push(invoice);
        this._model.updateBindings(true);
    }
}
```

UI5con

Inversion of Control using Dependency Injection

Helps for: Reusability, Extensibility

Some possible solutions

- Use service locator
- Use the `ServiceFactory` API
- Use ⚡ Dependency Injection ⚡

An overview of the landscape

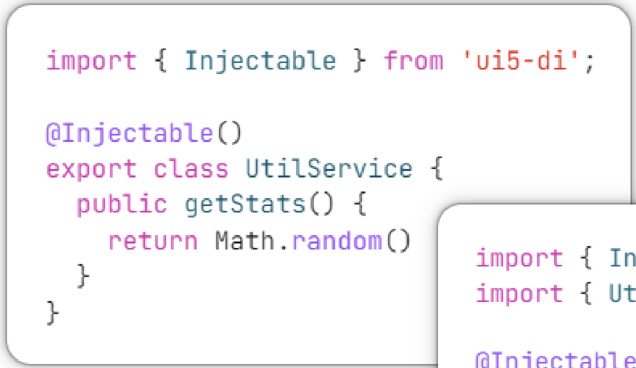
- React
- Angular & NestJS

UI5con

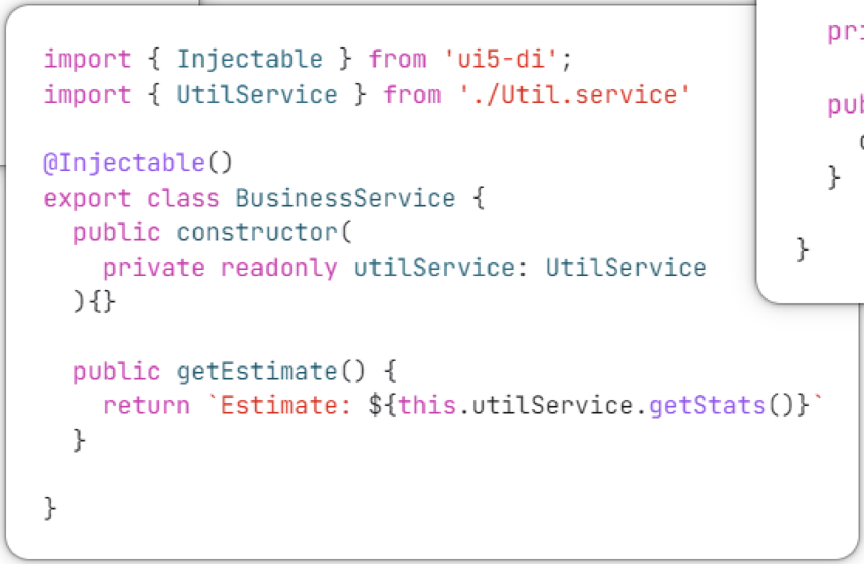
Inversion of Control using Dependency Injection

Helps for: Reusability, Extensibility

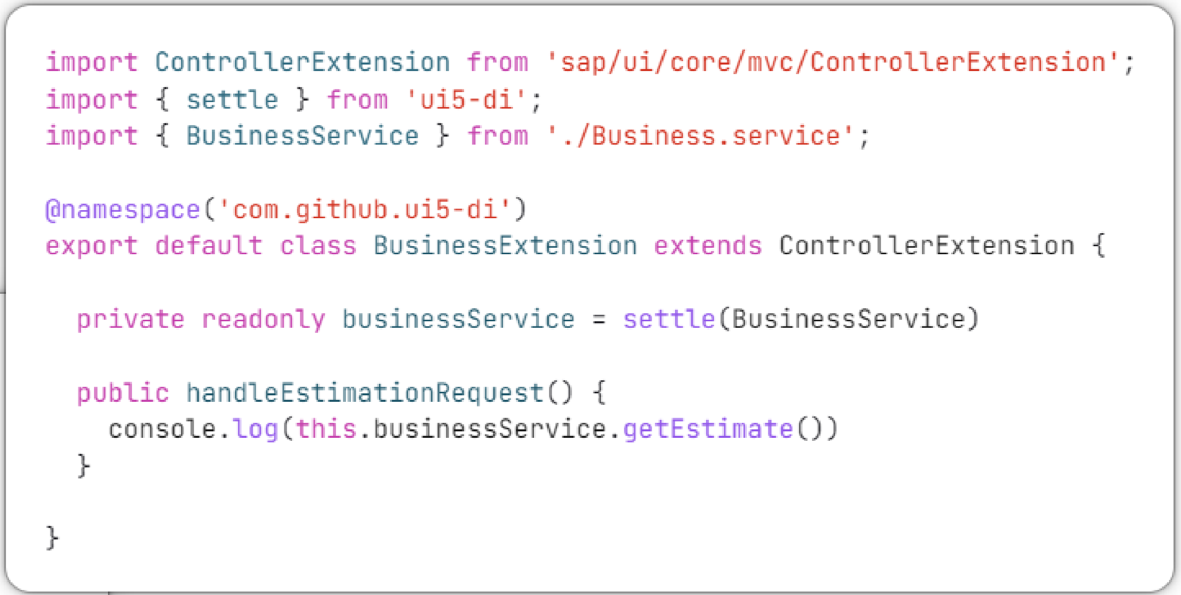
Feature Overview



```
import { Injectable } from 'ui5-di';

@Injectable()
export class UtilService {
  public getStats() {
    return Math.random()
  }
}
```



```
import { Injectable } from 'ui5-di';
import { UtilService } from './Util.service'

@Injectable()
export class BusinessService {
  public constructor(
    private readonly utilService: UtilService
  ){}

  public getEstimate() {
    return `Estimate: ${this.utilService.getStats()}`
  }

}
```



```
import ControllerExtension from 'sap/ui/core/mvc/ControllerExtension';
import { settle } from 'ui5-di';
import { BusinessService } from './Business.service';

@namespace('com.github.ui5-di')
export default class BusinessExtension extends ControllerExtension {

  private readonly businessService = settle(BusinessService)

  public handleEstimationRequest() {
    console.log(this.businessService.getEstimate())
  }

}
```

Like what you just saw?
Please rate my session in the event app:

UI5con

Thank you!

Dimitar Fenerski, sproutfinance.io

@dfenersky

https://github.com/dfenerski/ui5-state

https://github.com/dfenerski/ui5-di