Streamlining Contributions With EclipseTheia

paul.marechal@ericsson.com

EclipseCon Germany 2018

Talk

- Developing today
- Cloud Development
  - Eclipse Theia
- Theia Development
- Deployement
  - Workspace managers

Developing Today

EclipseTheia

Developing Today

- Checkout the project
- Install the tools / Configure the tools
- Install dependencies
- Start contributing
- Rinse and repeat (× projects × machines/users)

EclipseTheia

Developing Today - More

- Use the correct Operating System
- Dependency collisions between projects?
- Not your habitual workstation? Have fun recreating something of an environment.

What we would like to do

- Checkout the project
- Start contributing

EclipseTheia

What we would like to do

- Removing contribution overhead
- It is about changing habits and the ways of working
  - Develop anywhere
  - No compromises

EclipseTheia

Cloud Development

EclipseTheia

Not a new concept

- `vi` over `ssh` (remote development)
- Cloud9 (proprietary)
- GitLab IDE (minimalist)
- JDT/CDT over remote desktop (lag)
- ...

EclipseTheia

Theia

TypeFox

ERICSSON

redhat

arm

IBM

Theia Framework

- Written in TypeScript
- Runs in the browser
- Server-side
  - Node.js
- Runs on your desktop
  - Electron

EclipseTheia

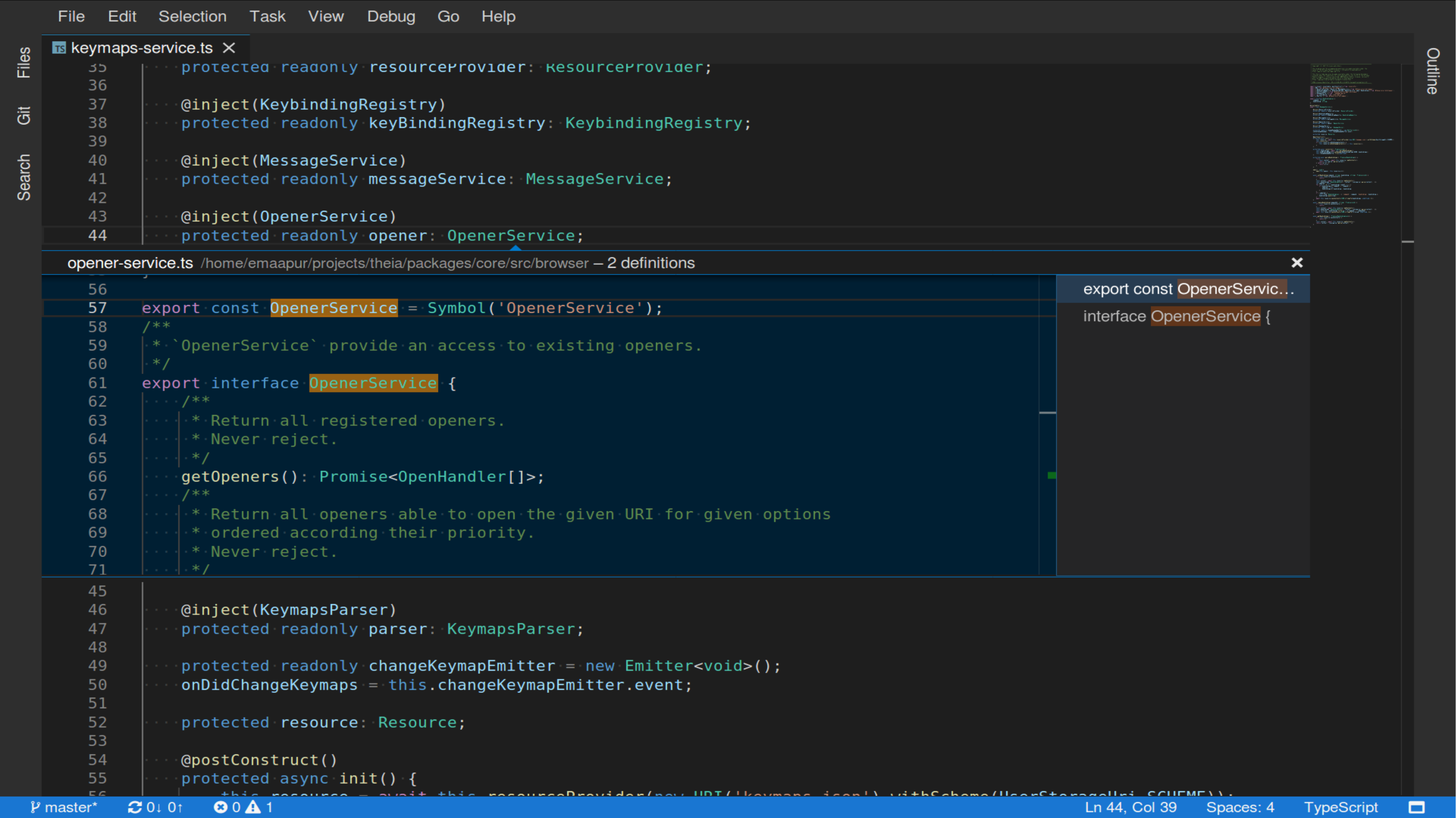
Theia Features

- Code navigation
  - Code completion, go to definition, ...
  - LSP based
- Debugging
  - Breakpoints, step over/in/out
  - DAP based

EclipseTheia

LSP/DAP?

- Language Server Protocol
  - Deep language understanding factorization
- Debug Adapter Protocol
  - Generic protocol to drive a debug session



```
    protected readonly resourceProvider: ResourceProvider;

    @inject(KeybindingRegistry)
    protected readonly keyBindingRegistry: KeybindingRegistry;

    @inject(MessageService)
    protected readonly messageService: MessageService;

    @inject(OpenerService)
    protected readonly opener: OpenerService;
```

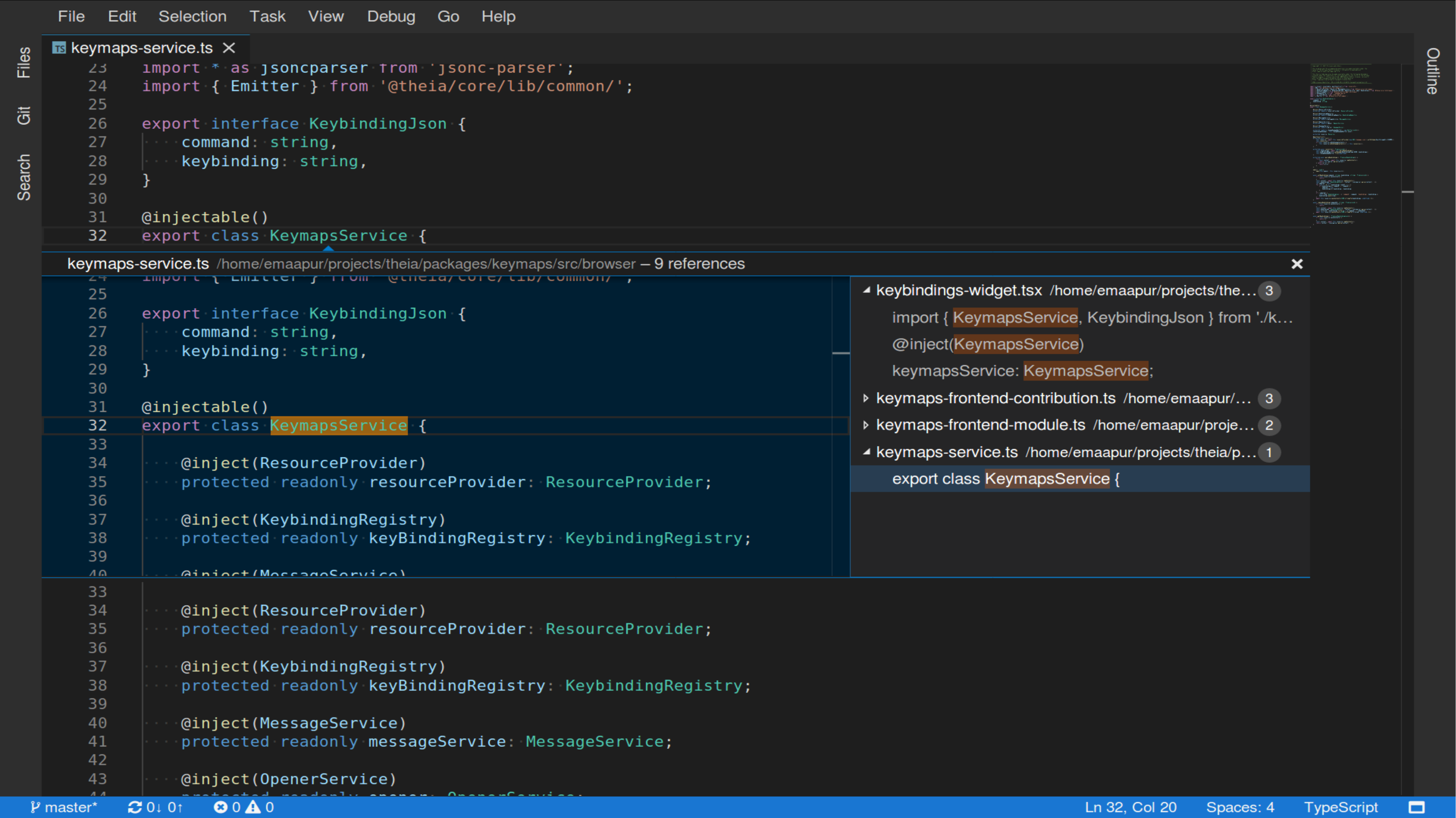
opener-service.ts /home/emaapur/projects/theia/packages/core/src/browser – 2 definitions

```

export const OpenerService = Symbol('OpenerService');
/**
 * `OpenerService` provide an access to existing openers.
 */
export interface OpenerService {
    /**
     * Return all registered openers.
     * Never reject.
     */
    getOpeners(): Promise<OpenHandler[]>;
    /**
     * Return all openers able to open the given URI for given options
     * ordered according their priority.
     * Never reject.
     */
```

export const OpenerServic...
interface OpenerService {

```

    @inject(KeymapsParser)
    protected readonly parser: KeymapsParser;

    protected readonly changeKeymapEmitter = new Emitter<void>();
    onDidChangeKeymaps = this.changeKeymapEmitter.event;

    protected resource: Resource;

    @postConstruct()
    protected async init() {
        this.resource = await this.resourceProvider(new URI('keymaps.json').withScheme(UserStorageUri.SCHEME));
```

master* 0↓ 0↑ 0 1 Ln 44, Col 39 Spaces: 4 TypeScript



```
import * as jsoncparser from 'jsonc-parser';
import { Emitter } from '@theia/core/lib/common/';

export interface KeybindingJson {
    command: string,
    keybinding: string,
}

@injectable()
export class KeymapsService {
```

keymaps-service.ts /home/emaapur/projects/theia/packages/keymaps/src/browser — 9 references

```
import { Emitter } from '@theia/core/lib/common/';

export interface KeybindingJson {
    command: string,
    keybinding: string,
}

@injectable()
export class KeymapsService {

    @inject(ResourceProvider)
    protected readonly resourceProvider: ResourceProvider;

    @inject(KeybindingRegistry)
    protected readonly keyBindingRegistry: KeybindingRegistry;

    @inject(MessageService)
```

```
keybindings-widget.tsx /home/emaapur/projects/the... 3
    import { KeymapsService, KeybindingJson } from './k...
    @inject(KeymapsService)
    keymapsService: KeymapsService;
keymaps-frontend-contribution.ts /home/emaapur/... 3
keymaps-frontend-module.ts /home/emaapur/proje... 2
keymaps-service.ts /home/emaapur/projects/theia/p... 1
    export class KeymapsService {
```

```

    @inject(ResourceProvider)
    protected readonly resourceProvider: ResourceProvider;

    @inject(KeybindingRegistry)
    protected readonly keyBindingRegistry: KeybindingRegistry;

    @inject(MessageService)
    protected readonly messageService: MessageService;

    @inject(OpenerService)
```

EclipseTheia

Theia Features

- Git support
- Terminal
- Tasks
- ...

EclipseTheia

Theia Framework

- Extensibility
  - Extensions
  - Plugins

EclipseTheia

Theia Application

- Composition of extensions
  - `$ npm install <theia-extension>`
- Two targets
  - "browser" (browser + Node.js server application)
  - "electron" (desktop application)

EclipseTheia

Theia Development

EclipseTheia

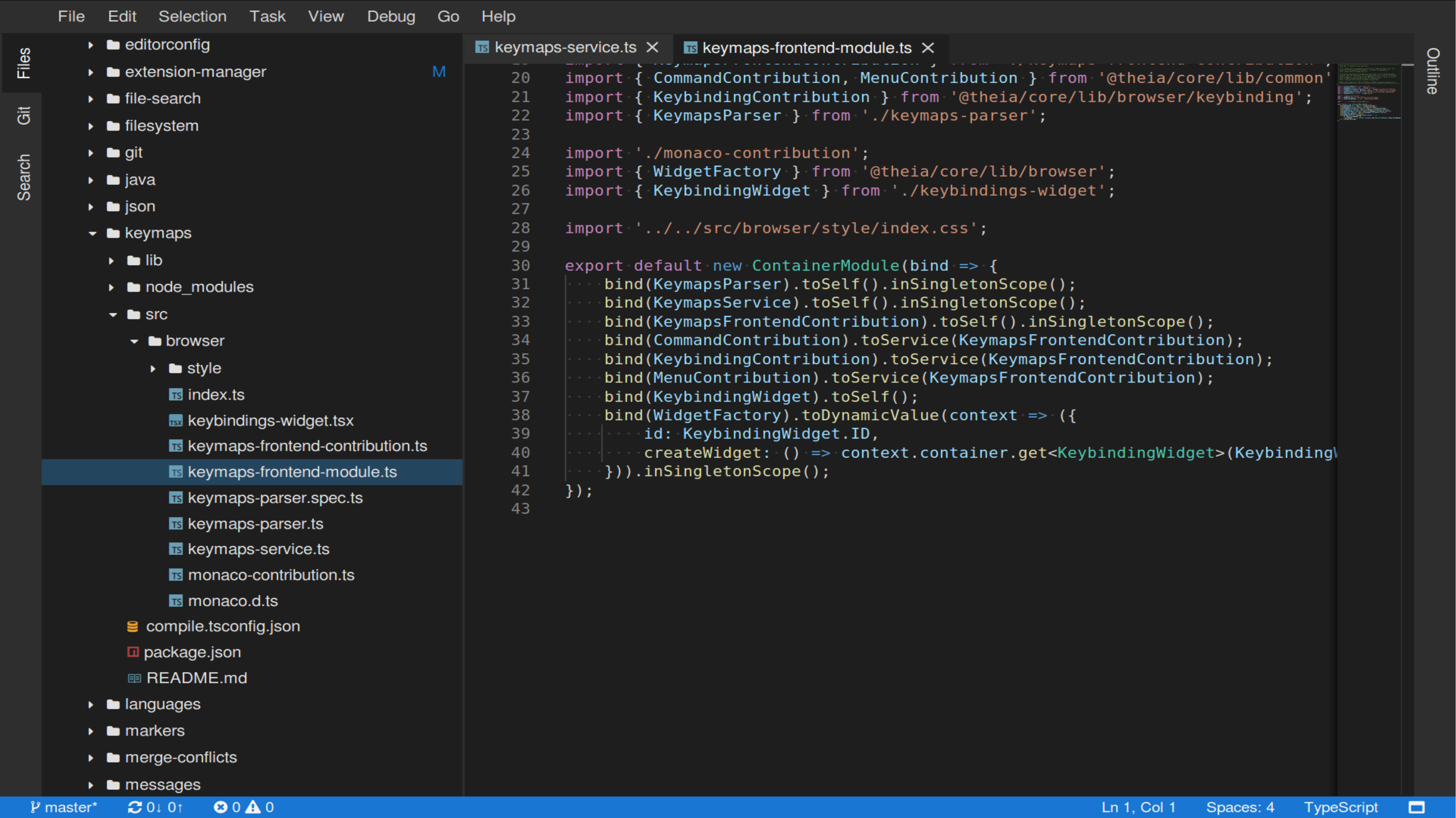
Theia Development

- Extensions (application build time)
  - Total control, rebinding of components, ...
- Plugins (runtime)
  - More rigid API via specific contribution points

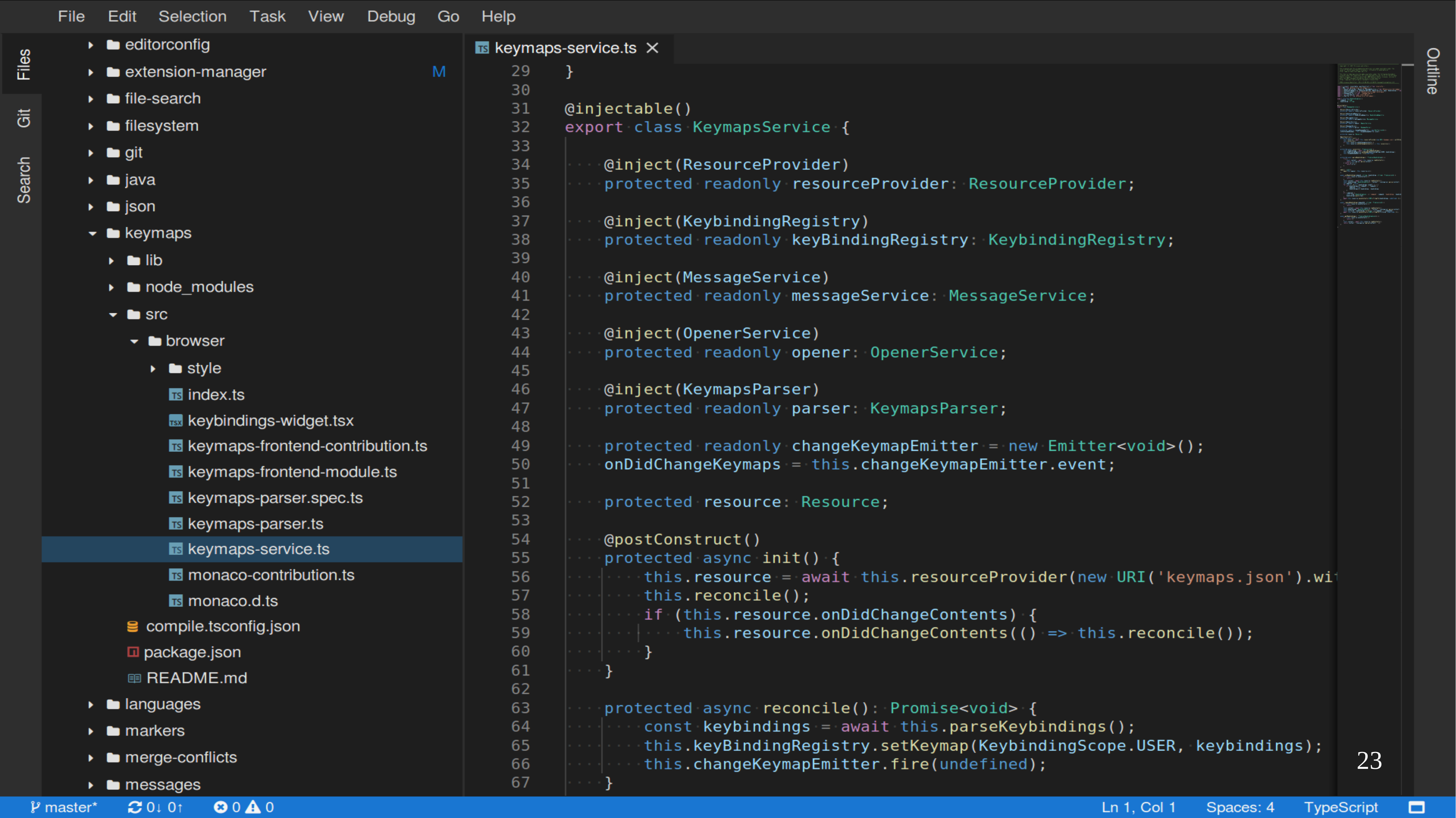
EclipseTheia

Theia Extensions

- Published as npm packages
  - `$ npm search @theia/*`
- Dependency injection
  - Add/Override any Theia component
  - InversifyJs



```
import { CommandContribution, MenuContribution } from '@theia/core/lib/common'
import { KeybindingContribution } from '@theia/core/lib/browser/keybinding';
import { KeymapsParser } from './keymaps-parser';

import './monaco-contribution';
import { WidgetFactory } from '@theia/core/lib/browser';
import { KeybindingWidget } from './keybindings-widget';

import '../../src/browser/style/index.css';

export default new ContainerModule(bind => {
    bind(KeymapsParser).toSelf().inSingletonScope();
    bind(KeymapsService).toSelf().inSingletonScope();
    bind(KeymapsFrontendContribution).toSelf().inSingletonScope();
    bind(CommandContribution).toService(KeymapsFrontendContribution);
    bind(KeybindingContribution).toService(KeymapsFrontendContribution);
    bind(MenuContribution).toService(KeymapsFrontendContribution);
    bind(KeybindingWidget).toSelf();
    bind(WidgetFactory).toDynamicValue(context => ({
        id: KeybindingWidget.ID,
        createWidget: () => context.container.get<KeybindingWidget>(KeybindingW
    })).inSingletonScope();
});

```



```
editorconfig
extension-manager        M
file-search
filesystem
git
java
json
keymaps
    lib
    node_modules
    src
        browser
            style
            index.ts
            keybindings-widget.tsx
            keymaps-frontend-contribution.ts
            keymaps-frontend-module.ts
            keymaps-parser.spec.ts
            keymaps-parser.ts
            keymaps-service.ts
            monaco-contribution.ts
            monaco.d.ts
    compile.tsconfig.json
    package.json
    README.md
languages
markers
merge-conflicts
messages
```

```typescript
}

@injectable()
export class KeymapsService {

    @inject(ResourceProvider)
    protected readonly resourceProvider: ResourceProvider;

    @inject(KeybindingRegistry)
    protected readonly keyBindingRegistry: KeybindingRegistry;

    @inject(MessageService)
    protected readonly messageService: MessageService;

    @inject(OpenerService)
    protected readonly opener: OpenerService;

    @inject(KeymapsParser)
    protected readonly parser: KeymapsParser;

    protected readonly changeKeymapEmitter = new Emitter<void>();
    onDidChangeKeymaps = this.changeKeymapEmitter.event;

    protected resource: Resource;

    @postConstruct()
    protected async init() {
        this.resource = await this.resourceProvider(new URI('keymaps.json').wi
        this.reconcile();
        if (this.resource.onDidChangeContents) {
            this.resource.onDidChangeContents(() => this.reconcile());
        }
    }

    protected async reconcile(): Promise<void> {
        const keybindings = await this.parseKeybindings();
        this.keyBindingRegistry.setKeymap(KeybindingScope.USER, keybindings);
        this.changeKeymapEmitter.fire(undefined);
    }
```

EclipseTheia

Theia Extensions

- Various contribution points for extenders
  - Menus/Commands/Openers
  - Views/Widgets
  - Language Clients
  - TextMate Grammars
  - And more...
- Everything in the framework can be made as an extension

EclipseTheia

Theia Plugins

- Self-contained (no dependency resolution)
- Runtime isolation
- Simple API

EclipseTheia

Theia Plugins

- API can be extended by Extensions
- VS Code extensions compatibility underway

EclipseTheia

Deployment

Cloud?

- Cloud development requires infrastructure
  - Networked machines
  - Administration (docker/kubernetes, ...)
  - Teams to operate

EclipseTheia

Docker

- Basic docker images available
  - github.com/theia-ide/theia-apps

EclipseTheia

Gitpod

- TypeFox cloud development platform
- Checkout PR branch and review it
- Theia-based

gitpod.io

EclipseTheia

Eclipse Che

- RedHat cloud workspace manager
- Heavy-duty
- Theia-able

www.eclipse.org/che

Personal Use

- Remote
  - Workspace manager
  - Vanilla Theia application instance
- Local
  - Electron application

EclipseTheia

Conclusion

EclipseTheia

What does Theia change?

- Allows anyone to have its own IDE accessible from a browser
- Workspace managers can expose the IDE to work in the space

EclipseTheia

Contributing now

- Contribute from (almost) anywhere
- Avoid setup overhead
- Focus on task at hand

EclipseTheia

EclipseTheia

Contacts

- Presenter
  - Paul Maréchal: paul.marechal@ericsson.com
- Social
  - gitter.im/theia-ide/theia
  - github.com/theia-ide/theia/issues

References

- github.com/theia-ide/theia
- eclipse.org/che
- gitpod.io

EclipseTheia

Cue and Hay

(Q&A)

eclipsecon Europe 2018

OSGi™ Community Event 2018

Evaluate the Sessions

Sign in and vote at **eclipsecon.org**

-1 0 +1