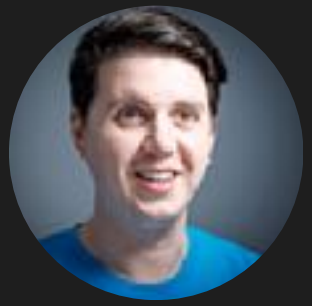
International JavaScript Conference

Angular Schematics

The Solution to All Your Problems

Viktor Slavov

@swagamemnon1



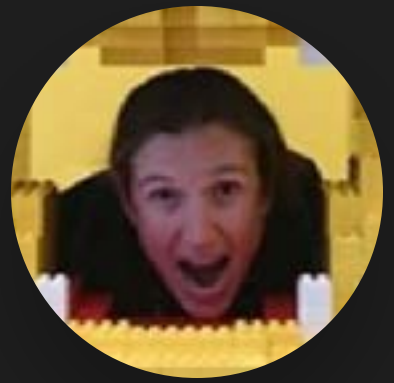
Damyan Petev

@damyanpetev

@infragistics

All Your Problems

- Initial project setup
- Adding new libraries
- Updates and breaking changes



@ErinJZimmer

Scott Hanselman
@shanselman

Replying to @tesseractiv

@tesseractiv @tjanczuk ah, whatever problem you have. If you have this problem, this solves it.

6:48 PM · Apr 30, 2014 · Twitter for iPhone

https://twitter.com/shanselman/status/461532471037677568

Breakdown

1. Schematics Intro
2. Authoring Schematics
3. Schematics API
4. Component Example
5. Angular CLI Hooks
6. TypeScript API

Schematics Intro

Schematics Intro

What are schematics

- Building block for the Angular CLI
- Make transformations on an in-memory representation of the file system (`Tree`)
  - Performance, --dry-run, advanced operations (merge)
- Define highly adaptable functions (`Rule`)
- Ability to ensure modularity and testability (`collection.json`)

```
> ng g c my-component

> ng add my-favorite-library

> ng new my-angular-project

> ng update my-outdated-library@latest
```

Some schematics we use daily

Authoring Schematics

Quickly get started with the schematics cli

- `npm install -g @angular-devkit/schematics-cli`
- `schematics blank --name=intro-schematic`

Schematic structure

- `package.json` - provide path to schematic collection
- `collection.json` - project's schematic collection
- `index.ts` - schematic factory

Authoring Schematics

```
  "license": "MIT",
  "schematics": "./src/
collection.json",
  "dependencies": {
```

Schematics property in `package.json`

```
{
    "$schema": "../node_modules/@angular-devkit/schematics/collection-
schema.json",
    "schematics": {
        "intro-schematic": {
            "description": "A blank schematic",
            "factory": "./intro-schematic/index#introSchematic"
        }
    }
}
```

An example schematics `colleciton.json`

Authoring Schematics

```
import { Rule, SchematicContext, Tree } from '@angular-devkit/
schematics';

export function introSchematic(_options: any): Rule {
  return (tree: Tree, _context: SchematicContext) => {
    _context.logger.info('Thanks for using schematics!');
    return tree;
  };
}
```

$schematic definition

Schematics API

Schematics API

What you can do

- Manipulate tree files/directories
- Context can schedule tasks and log messages
- Chain rules, merge tree states

Example scenario

- Log message (context API)
- Create config file (tree file manipulation API)
- Chain task execution (context API)

```
Thank you for installing my-lib!
CREATE custom-lib-config.json (85 bytes)
```

Schematics API

Modularity and reusability

```
"config": {
  "description": "Adds a custom .json file to the current workspace.",
  "factory": "./config/index"
},
"greeter": {
  "description": "Displays a 'Thank you' message",
  "factory": "./greeter/index"
},
"combined": {
  "description": "Displays a message and creates a file",
  "factory": "./combined/index"
}
```

Schematics API

Tree API

- Read files with tree.read()
- Create files with tree.create()

```
import { Rule, SchematicContext, Tree, SchematicsException } from '@angular-devkit/
schematics';

const config = {
    version: '1.0.0',
    workspace: ''
};

export default function config(_options: any): Rule {
    return (tree: Tree, _context: SchematicContext) => {

        if (!tree.exists('/package.json')) {
            throw new SchematicsException('Could not find package.json!');
        }

        const packageJSONContent = tree.read('/package.json').toString();

        const packageJSON = JSON.parse(packageJSONContent);
        const projectName = packageJSON.name;
        const fileContents = Object.assign({}, config, { workspace: projectName });
        tree.create('./custom-lib-config.json', JSON.stringify(fileContents, null, 4));
    };
}
```

Schematics API

Context API

- Logger
- Can add execution tasks
- 3 built-in tasks – schematic, git, package-install

```
import { Rule, SchematicContext, Tree } from '@angular-devkit/
schematics';

export default function greeter(_options: any): Rule {
  return (tree: Tree, context: SchematicContext) => {
    context.logger.info('Thanks for using schematics!');
    return tree;
  };
}
```

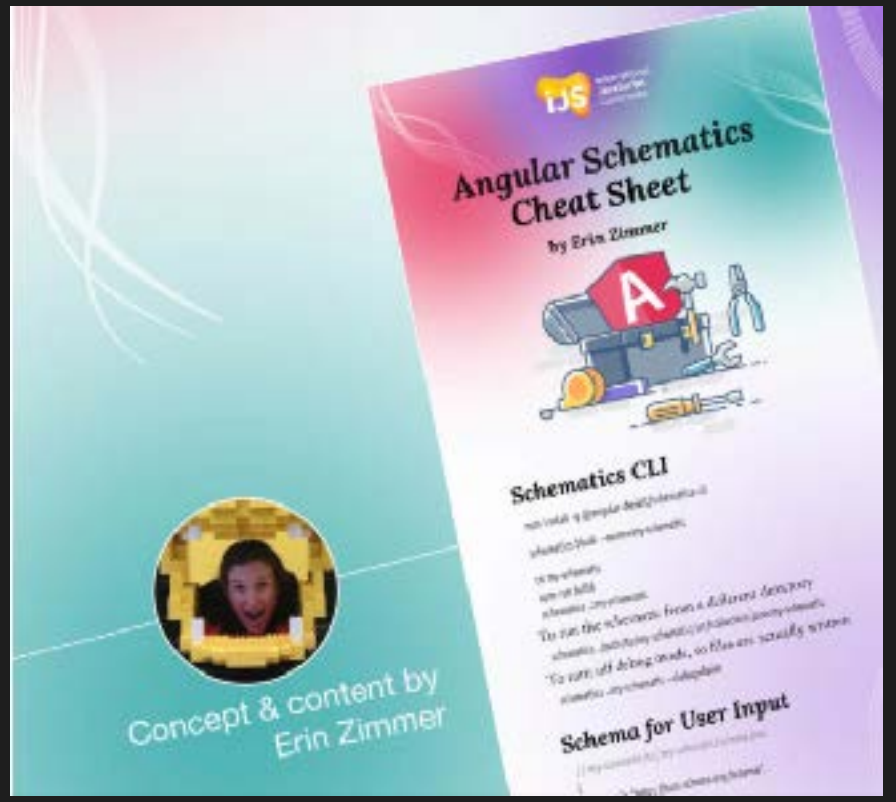
```
import { RunSchematicTask } from '@angular-devkit/schematics/tasks';

export default function combined(_options: any): Rule {
  return (tree: Tree, context: SchematicContext) => {
    const greeterTask = new RunSchematicsTask('greeter', {});
    const configTask = new RunSchematicsTask('config', {});

    const greeterTaskID = context.addTask(greeterTask);
    context.addTask(configTask, [greeterTaskID]);
    return tree;
  };
}
```

Angular Schematics Cheat Sheet

Get the most out of schematics using the [Angular Schematics Cheat Sheet](#) by Erin Zimmer:



Component Example

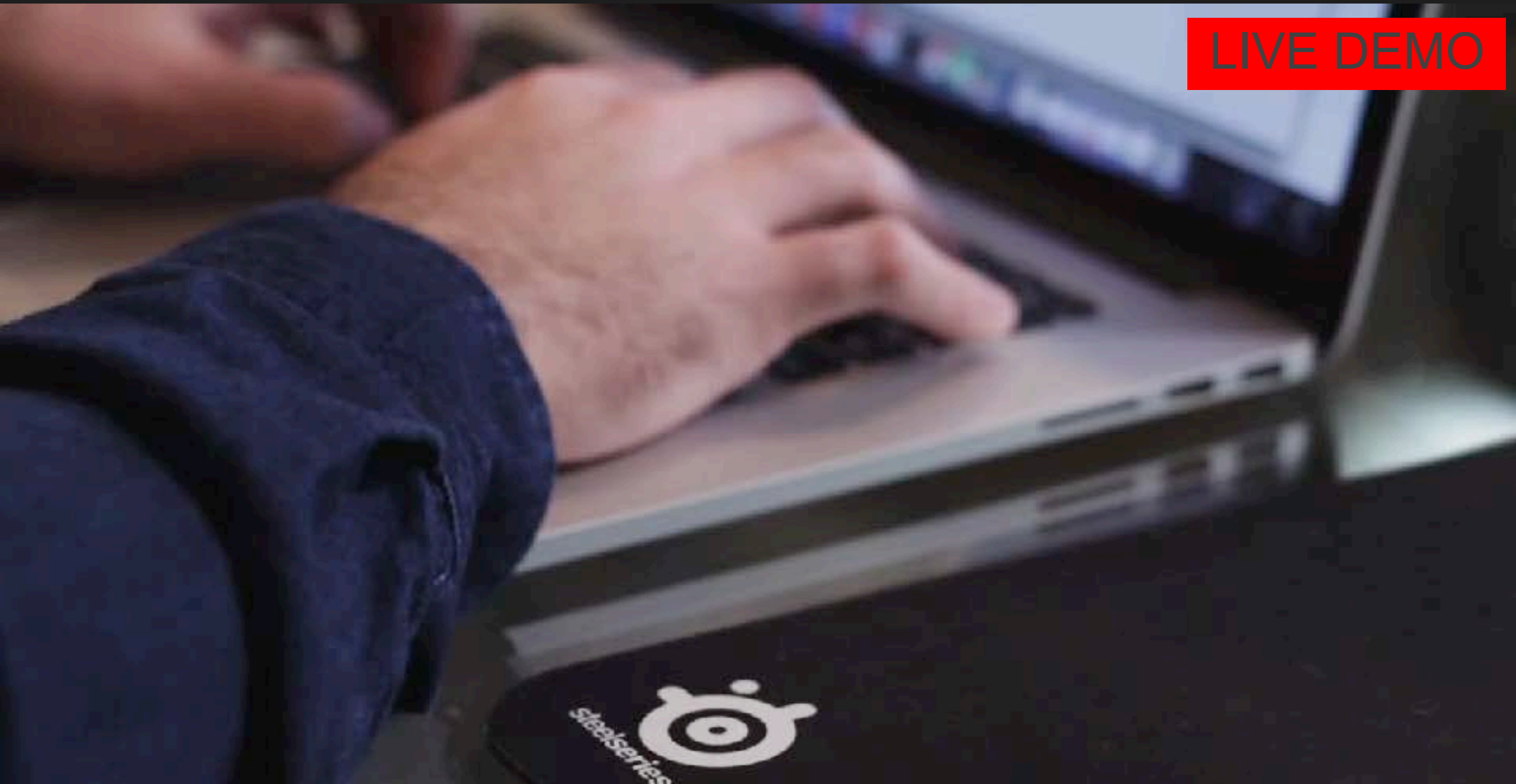
Component Example

Premise

- We'll go over an example of how to create a schematic that adds your custom component to the consuming project
- We'll take advantage of the following:

1. Strictly typing options via schema.json
2. Calling external schematic to create a component
3. Using schematics templating API to apply changes to component template

```
>ng g intro-schematic:component --name=my-card --type=card
```


LIVE DEMO

Angular CLI Hooks

ng new hook

ng-new schematic

- Can be ran automatically when *creating* a new project
- Requires a collection to be passed
- Can take care of all initial project setup

```
{
    "schematics": {
        "ng-new": {
            "description": "Creates a custom project",
            "factory": "./ng-new/index"
        },
```

```
>ng new --collection=intro-schematic my-project
```

`ng add` + `ng update`

Like npm's `postinstall`, but better

ng add hook

ng-add schematic

- Run your schematic automatically when your library is added to a project
- Easily onboard new users and minimize possibility of errors in initial configuration

```
{
    "schematics": {
        "ng-add": {
            "description": "Takes care of initial setup",
            "factory": "./ng-add/index"
        },
```

```
$ ng add igniteui-angular
Installing packages for tooling via npm.
```

`ng update` migrations

What are migrations

- Migrations are also Angular schematics
- Automatically run when calling `ng update [library-name][@version]`
- Extremely useful for both users and developers – easily handle breaking changes and deprecations

```
>ng update igniteui-angular@latest
```

`ng update` migrations

Defining migrations

- Separate collection
- Include path to collection in `ng-update` property in package.json
- Define `version` for each migration schematic to determine its migration scope

```
  "ng-update": {
    "migrations": "./migrations/migration-
collection.json"
  }
```

package.json

```
{
    "schematics": {
        "migration-1": {
            "version": "2.0.0",
            "description": "Migrates from version 1.0.0 to version 2.0.0",
            "factory": "./migration-1/index"
        },
```

migration-collection.json

`ng update` migrations

projects ▸ igniteui-angular ▸ migrations ▸ update-6_3 ▸ changes ▸ {} outputs.json ▸ [ ] changes ▸ {} 0 ▸ {} owner ▸ ⋯

```
{
    "$schema": "../../common/schema/binding.schema.json",
    "changes": [
        {
            "name": "onChange",
            "replaceWith": "onChanged",
            "owner": {
                "selector": "igxToggle",
                "type": "directive"
            }
        }
    ]
}
```

https://github.com/IgniteUI/igniteui-angular/wiki/Update-Migrations#using-the-updatechanges-class

TypeScript ❤ you!

TypeScript Compiler API

- With the TypeScript API you can parse files and do more precise corrections
- Typed access to constructs such as `Program` and `SourceFile`

- You get full access to the TypeScript Abstract Syntax Tree (AST)
  - Declarations, imports, variables, types... ALL the things

```
(property) ts.SourceFile.statements: ts.NodeArray<ts.Statement>
const imports = source.statements.filter((a: ts.Statement) => a is ts.ImportDeclaration>namedImportFilter);
```

- Add correct import declaration, property definitions and decorators

TypeScript Language Service

- Even more powerful capabilities:
- Trace variable references, find rename locations

```
const renames = service.findRenameLocations(sourcePath, elem.name.getStart(), false, false, false);
```

- Find definitions, matching braces, check for errors, etc

```
service.        You, a few seconds ago • Uncommitted
```

- findRenameLocations
- getApplicableRefactors
- getBraceMatchingAtPosition
- getBreakpointStatementAtPosition
- getCodeFixesAtPosition
- getCombinedCodeFix
- getCompilerOptionsDiagnostics
- getCompletionEntryDetails
- getCompletionEntrySymbol
- getCompletionsAtPosition
- getDefinitionAndBoundSpan
- getDefinitionAtPosition

Language Service with 🌳

```
const servicesHost: ts.LanguageServiceHost = {
    getCompilationSettings: () => options,
    getScriptFileNames: () => filePaths,
    getScriptVersion: fileName => {…
    },
    getScriptSnapshot: fileName => {
        if (!tree.exists(fileName)) {
            return undefined;
        }
        return ts.ScriptSnapshot.fromString(tree.read(fileName).toStri
ng());
    },
    getCurrentDirectory: () => process.cwd(),
    getDefaultLibFileName: opts => ts.getDefaultLibFilePath(opts),
    fileExists: fileName => {…
    }
};
```

```
const fileVersions = new Map<string, num
>();
patchTreeOverwrite(tree, fileVersions);
```

https://s

Save Changes to TypeScript files

- Depending on your needs
- Can do string manipulation – every node has start/end index
- Can manipulate the AST and print it back*

```
const source: ts.SourceFile = ts.createSourceFile(filePath, fileSource, ts.ScriptTarget.Latest, true);
const printer: ts.Printer = ts.createPrinter();
let text = printer.printFile(source);
```

```
const textChanges = services.getFormattingEditsForDocument(filePath, this.getFormattingOptions());
```

ng new


WRAP UP.

Resources

Ignite UI for Angular

https://www.infragistics.com/products/ignite-ui-angular

Docs

- Schematics overview
- Schematics README
- Using the TypeScript Compiler API

GitHub

- schematics-component-example repo
- igniteui-cli schematic implementation
- igniteui-angular schematic implementation

Infragistics

- Angular schematics for libraries
- Getting started with igniteui-cli and schematics
- Ignite UI for Angular Trial

Questions

Viktor Slavov

@swagamemnon1

Damyan Petev

@damyanpetev

INFRAGISTICS

iJS International JavaScript Conference