

SPFx CI/CD Pipelines on Azure DevOps

Serdar Ketenci
Architect, Simternet Inc

@serdarktnici

ketenci.serdar@hotmail.com

<https://www.serdarketenci.com>

Sponsors

Main Sponsor



Platinum Sponsor



Venue Sponsor



Silver Sponsor



Agenda

- Manual steps for shipping your solutions
- Demo
- CI (Continuous Integration)
- Demo
- Task Group
- Demo
- CD (Continuous Delivery)
- Demo
- Q&A

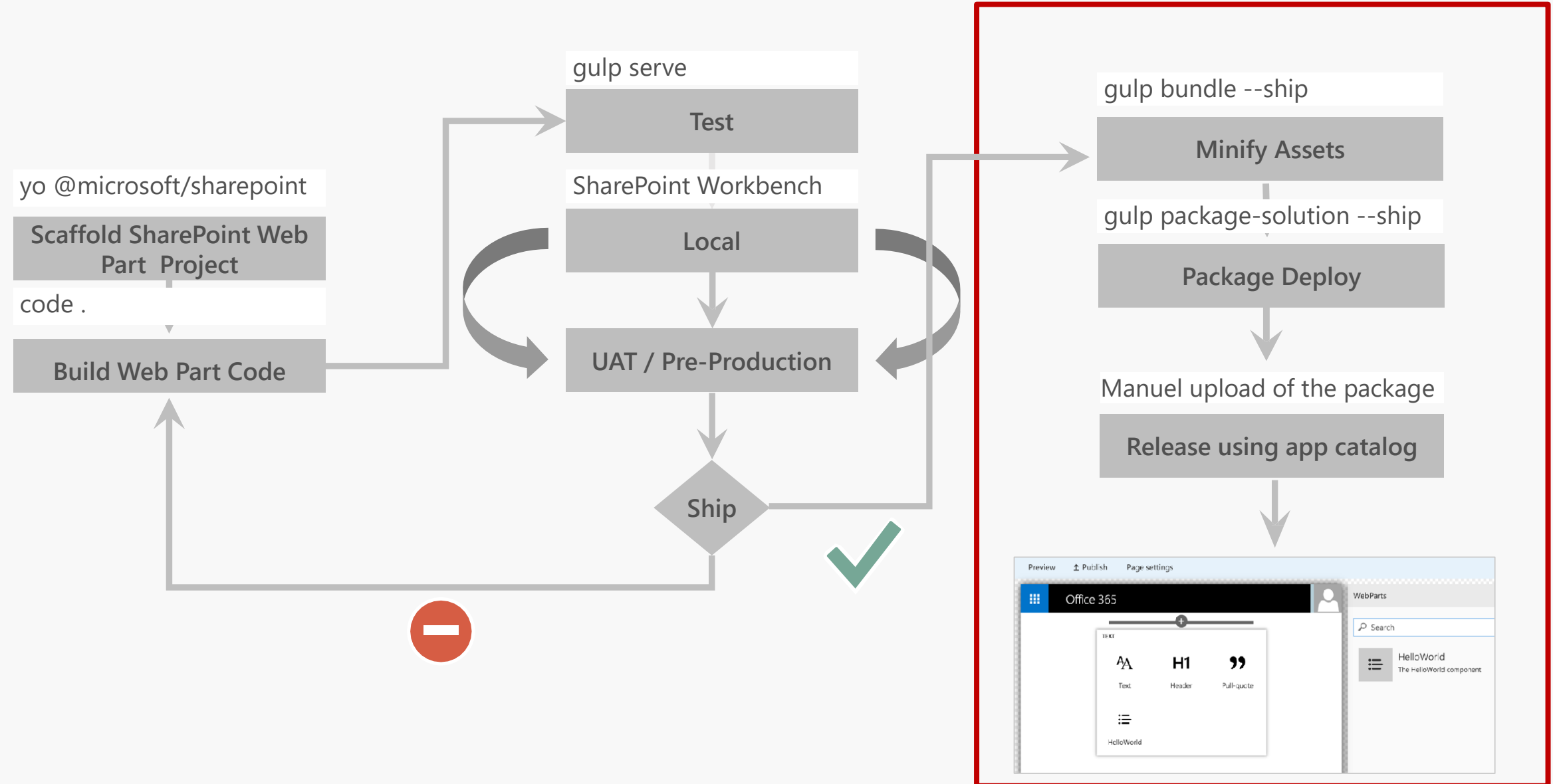


The project



Code is ready,
let's ship it!

Client-side solution creation flow



Manual steps for shipping your solutions

- Clean your solution
 - `gulp clean`
- Bundle your solution
 - `gulp bundle --ship`
- Package your solution
 - `gulp package-solution --ship`
- (Optionally) upload your files to your CDN
- Upload the solution package and deploy it in the app catalog (site or tenant)

`gulp clean`

Cleans project



`gulp bundle --ship`

Minify Assets



`gulp package-solution --ship`

Package Deploy



Manuel upload of the package

Release using app catalog



**Available on Classic and
Client-Side Pages**

A wooden arrow pointing to the right, mounted on a wooden post. The arrow is made of light-colored wood and has a simple, hand-drawn style. The background is blurred, showing some greenery and a building. An orange semi-transparent rectangle is overlaid on the left side of the arrow, containing the text "What are the options?" in white.

What are the
options?

Release process automation options

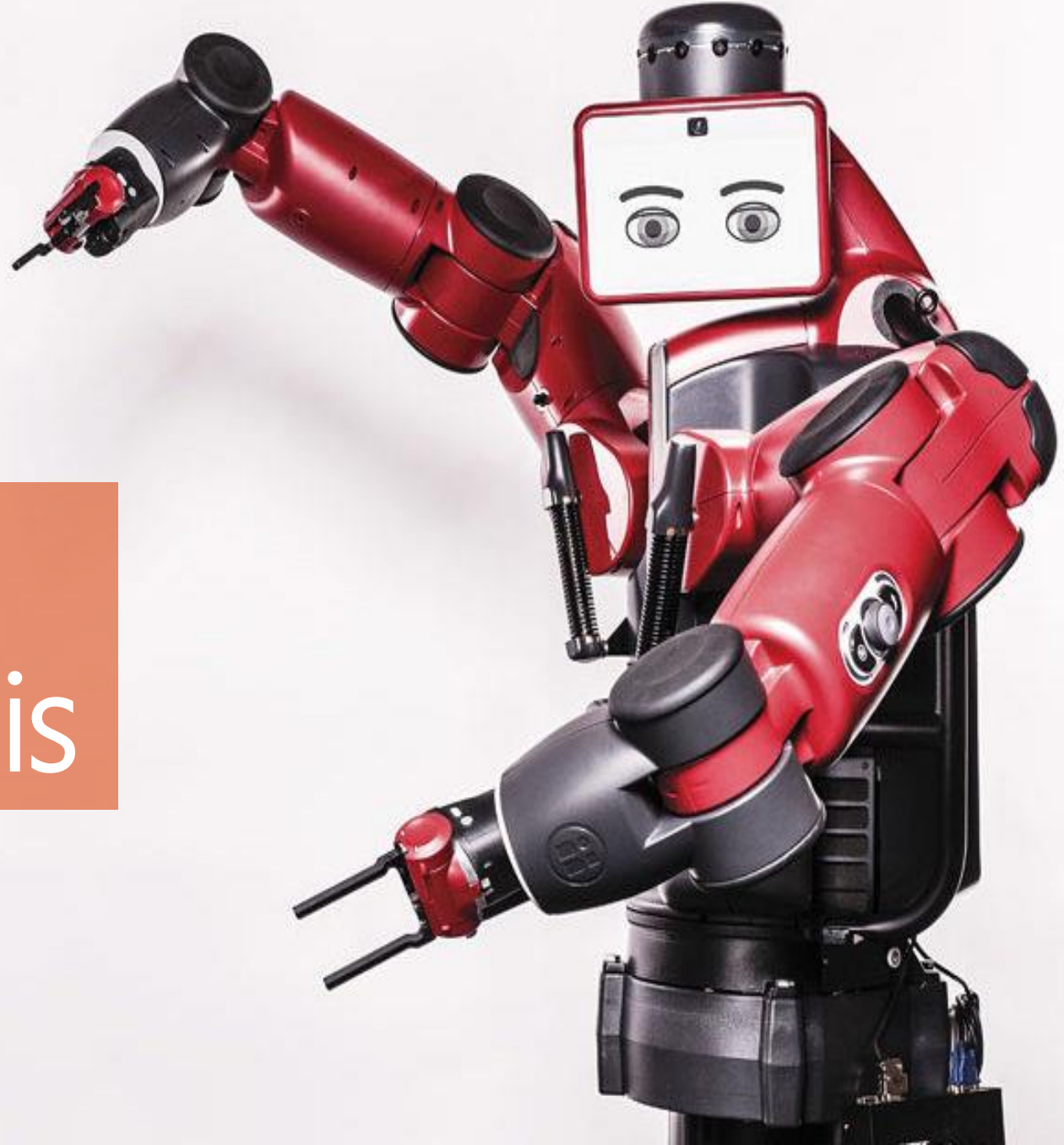
- Node script
- Gulp
- Office 365 CLI
- PowerShell
- VSCode extension
- ...





Demo: Manual steps

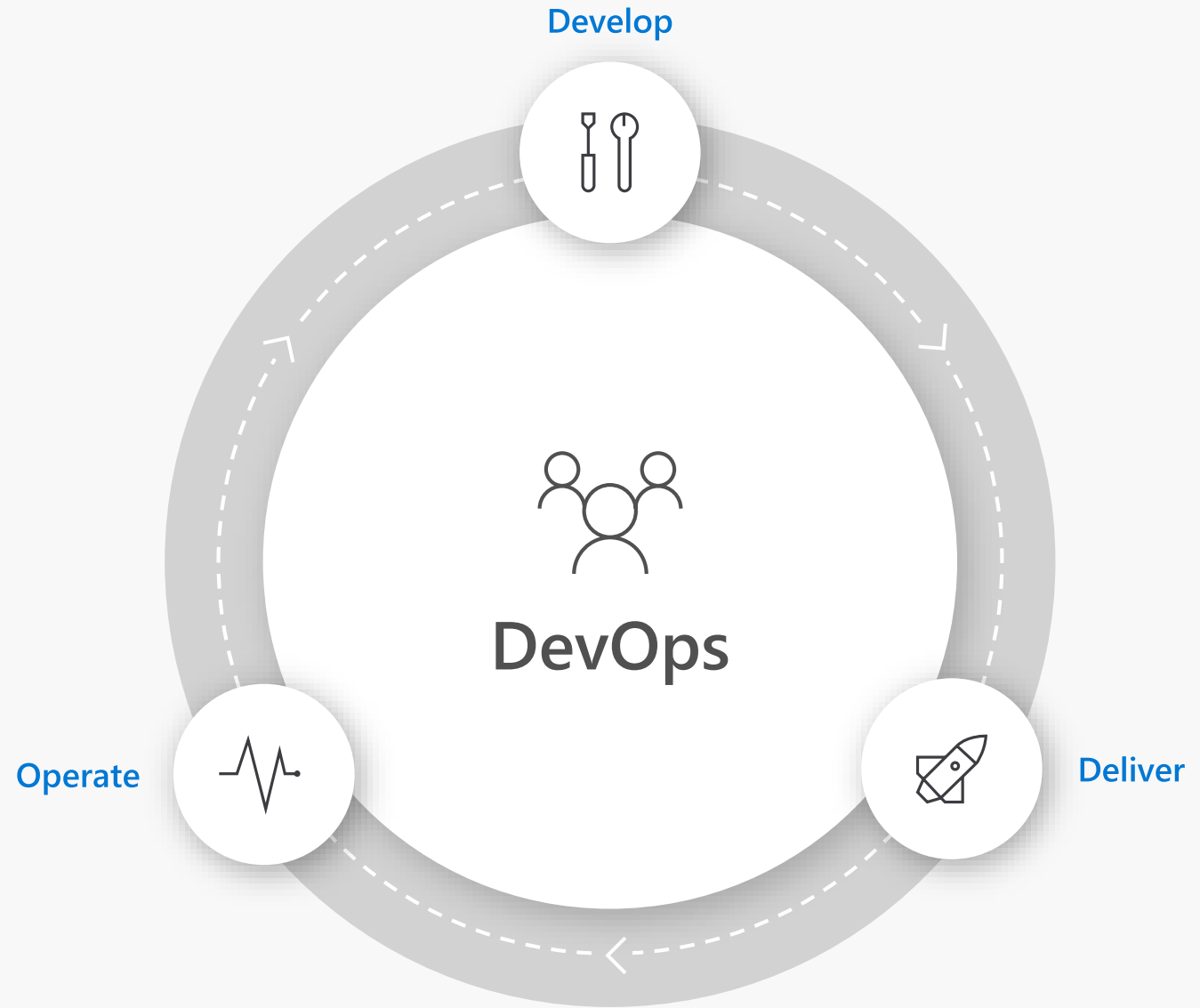
Gain time by
automating this



Accelerating Delivery with DevOps



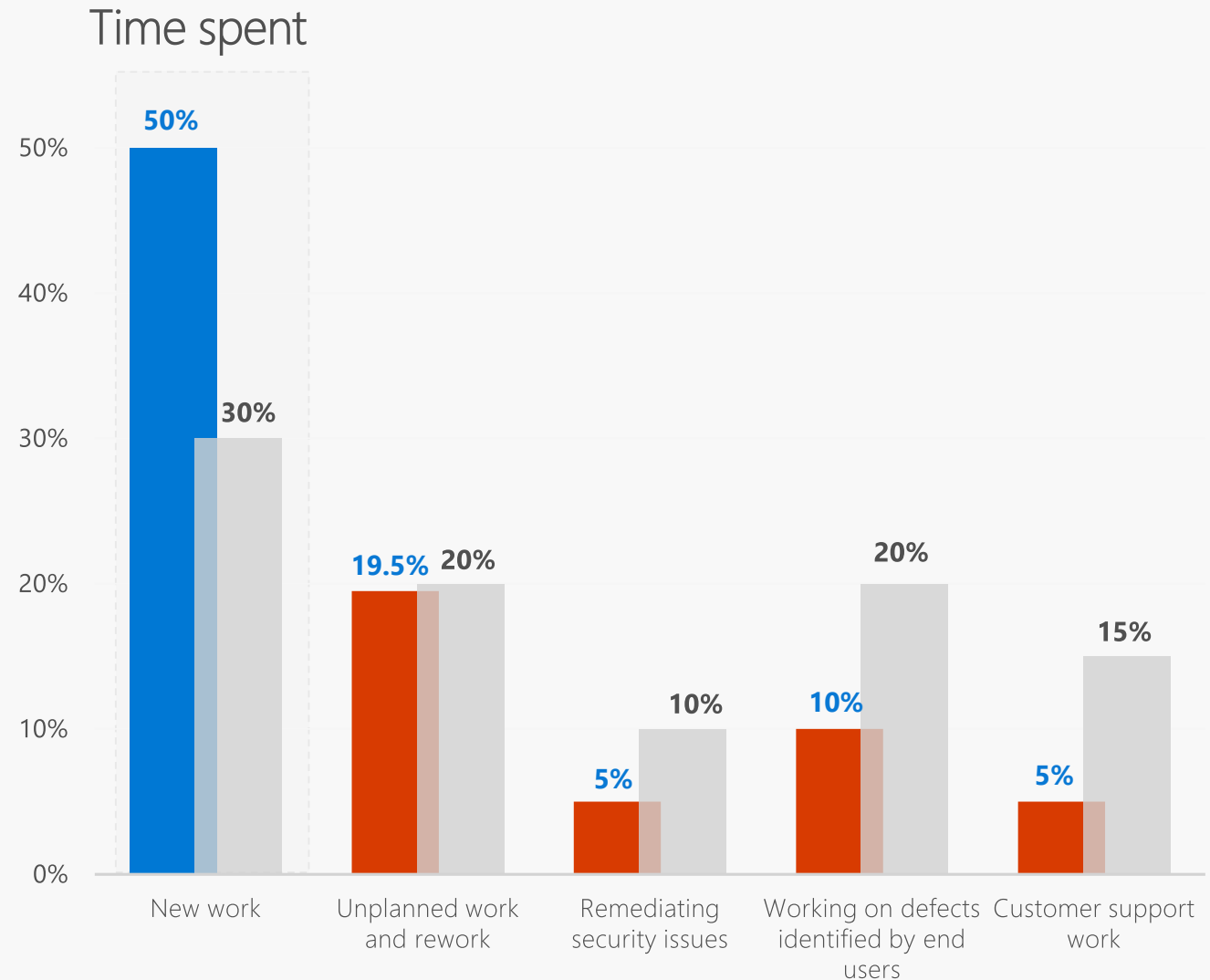
- DevOps is the union of **people**, **process**, and **products** to enable continuous delivery of value to your end users.



Innovation with oversight

Top performing DevOps companies spend more time innovating and less time “keeping the lights on”.

The result: better products, delivered faster, to happier customers by more engaged teams





Deliver with Azure DevOps



Azure Boards



Azure Repos



Azure Pipelines



Azure Artifacts



Azure Test Plans



Deliver



Azure Boards



Azure Repos



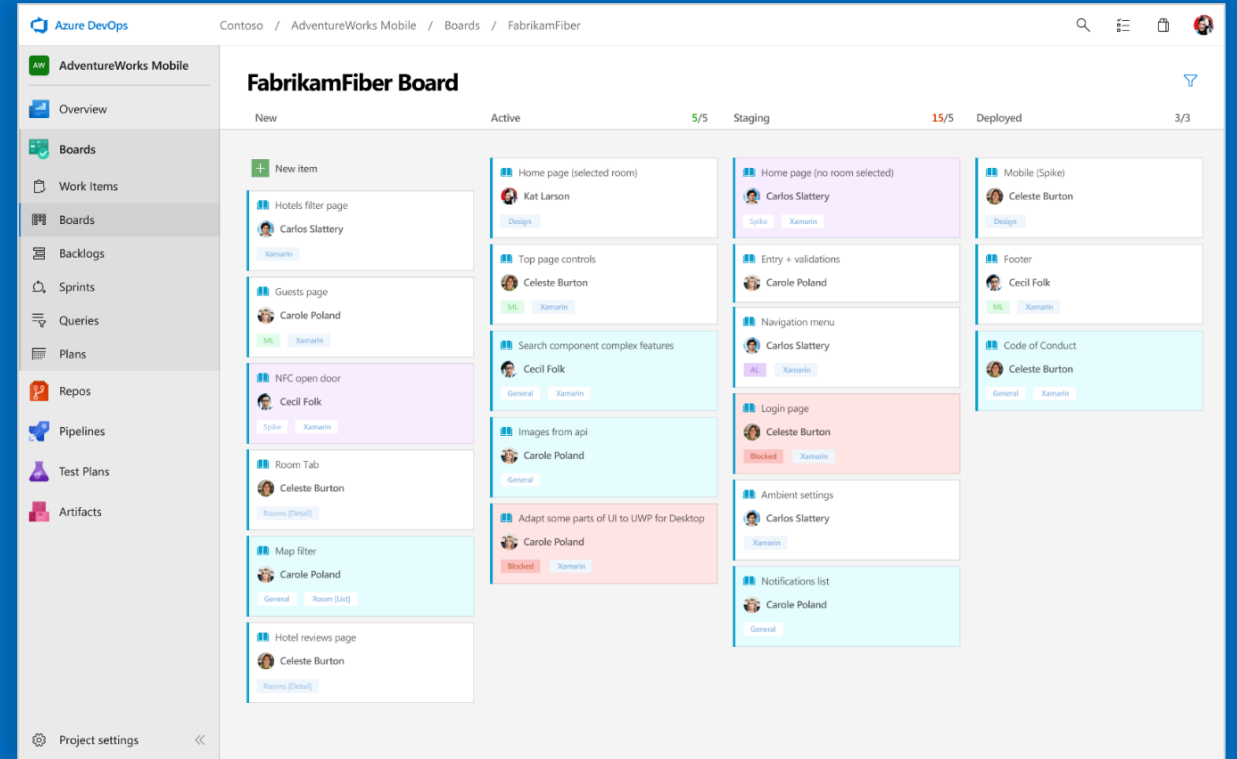
Azure Pipelines



Azure Artifacts



Azure Test Plans

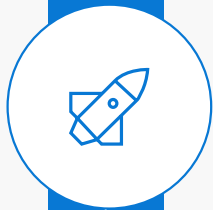


Connecting ideas to releases

Scrum ready to help your teams run sprints, stand-ups, and plan work

Integrated with GitHub commits and pull requests

Insights into project status and health



Deliver



Azure Boards



Azure Repos



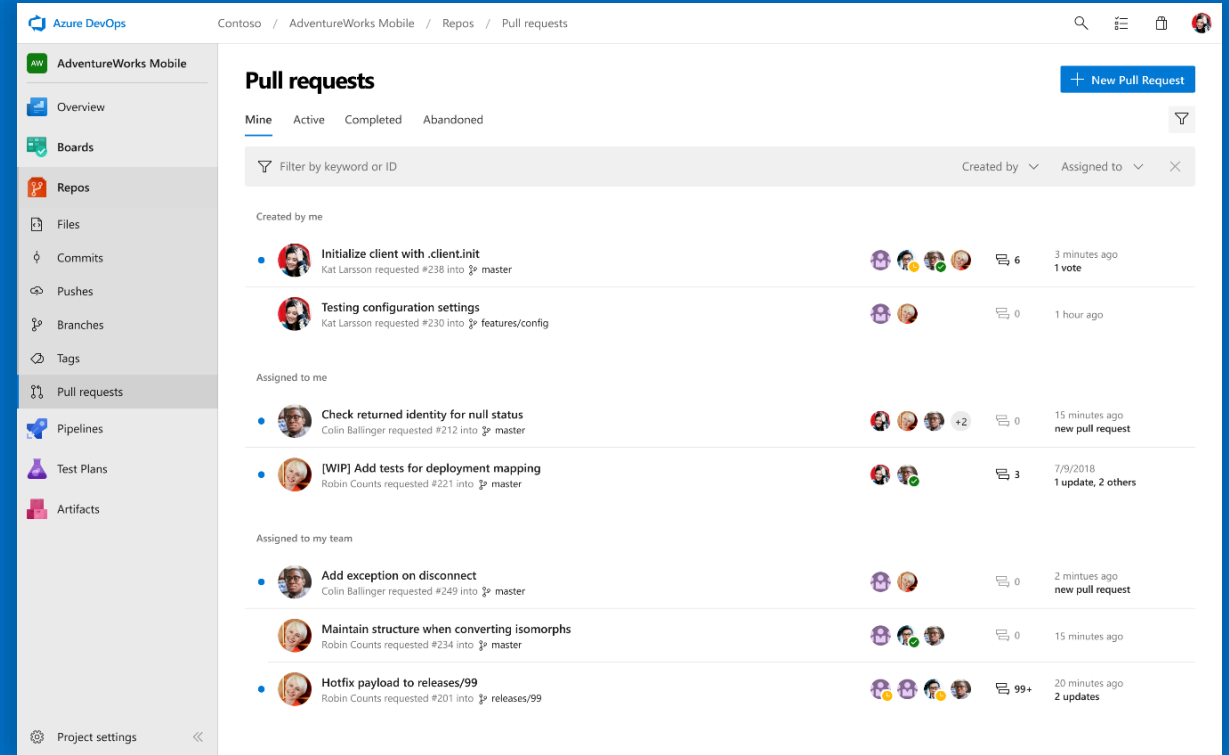
Azure Pipelines



Azure Artifacts



Azure Test Plans



Private Git and TFVC repos for your teams

Code review via branch pull requests

Branch policies and build validation

Easy migration path to / from GitHub



Deliver



Azure Boards



Azure Repos



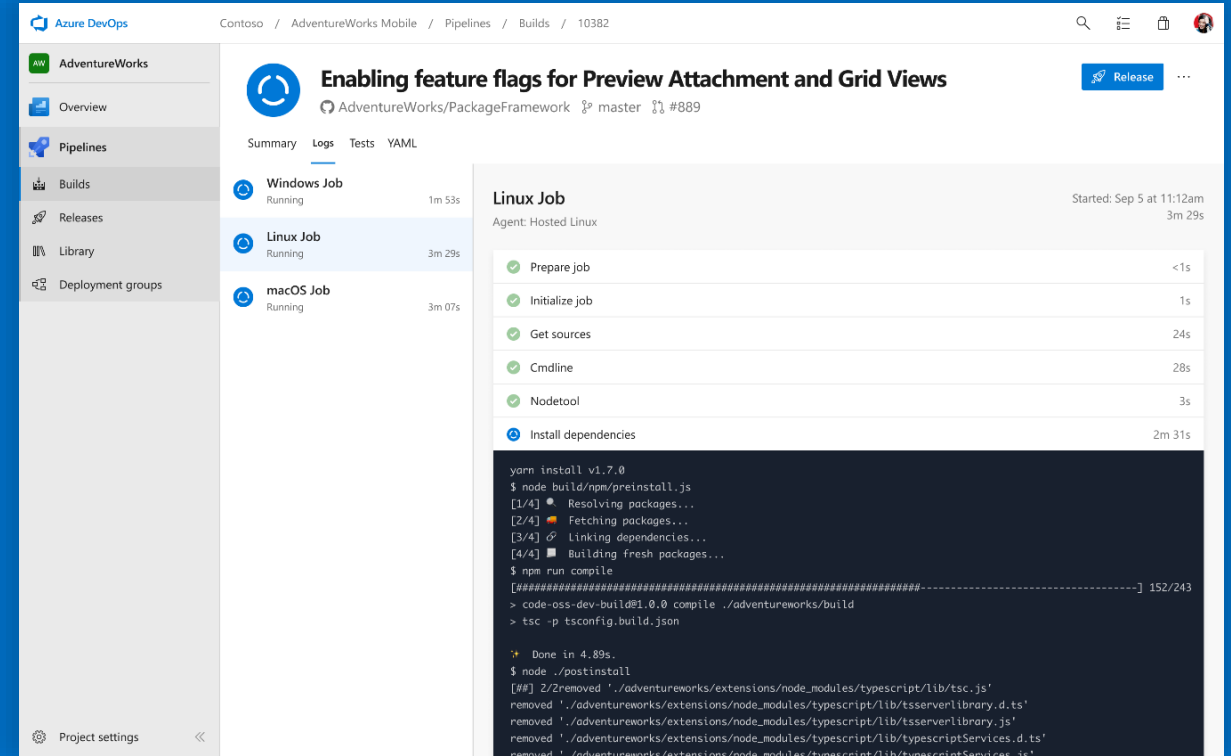
Azure Pipelines



Azure Artifacts



Azure Test Plans



Cloud-hosted pipelines for Linux, macOS and Windows

Any language, any platform, any cloud

Native support for containers and Kubernetes

Best-in-class for open source





Deliver



Azure Boards



Azure Repos



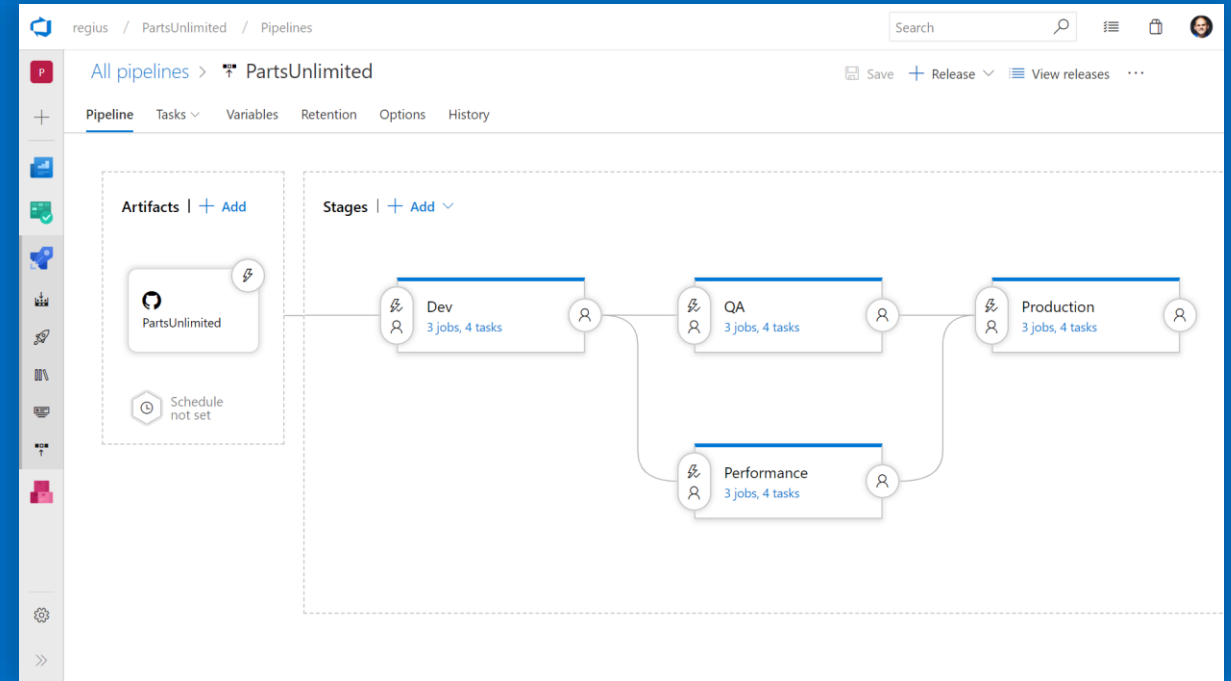
Azure Pipelines



Azure Artifacts



Azure Test Plans



Deploy to on-premises, ANY cloud or a hybrid of cloud and on-prem

Staged environment releases

Pre and post deployment approvals with gates to automate approval based on conditions



Deliver



Azure Boards



Azure Repos



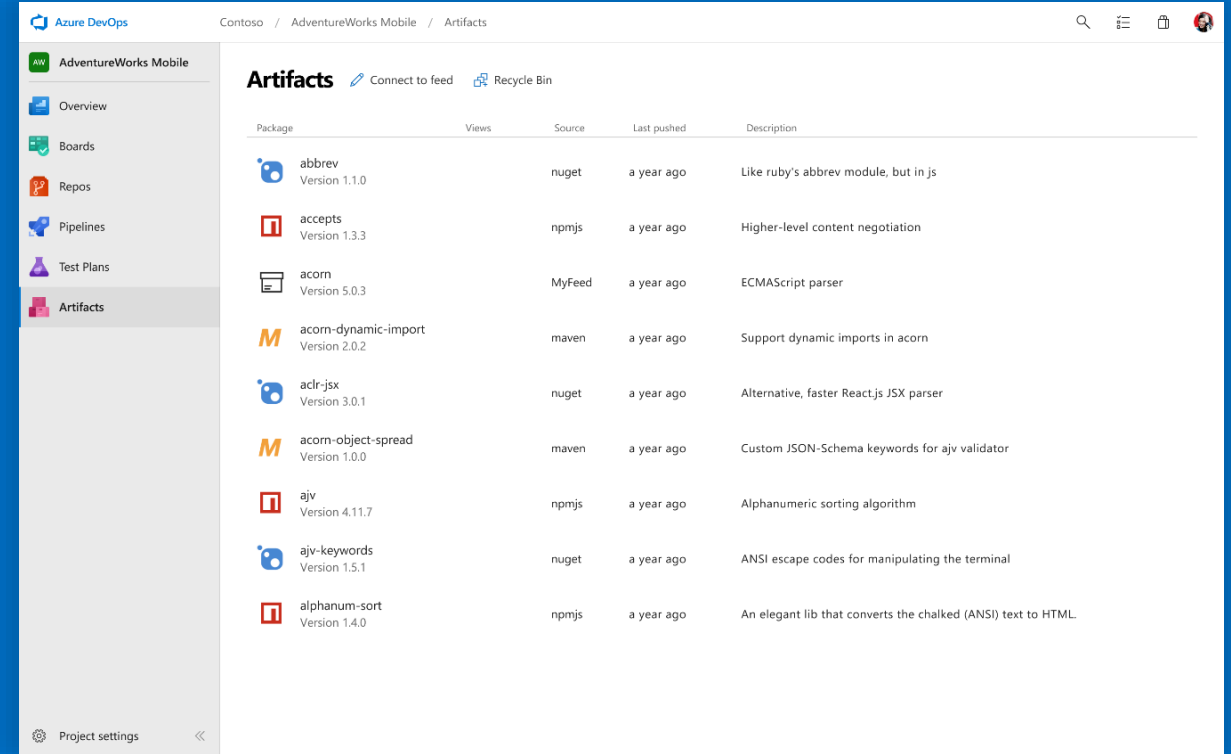
Azure Pipelines



Azure Artifacts



Azure Test Plans



Share code efficiently

Keep your Maven, npm, NuGet and Python packages and more in the same place

Aggregate from public registries and internal teams

Publish and track from any pipeline



Deliver



Azure Boards



Azure Repos



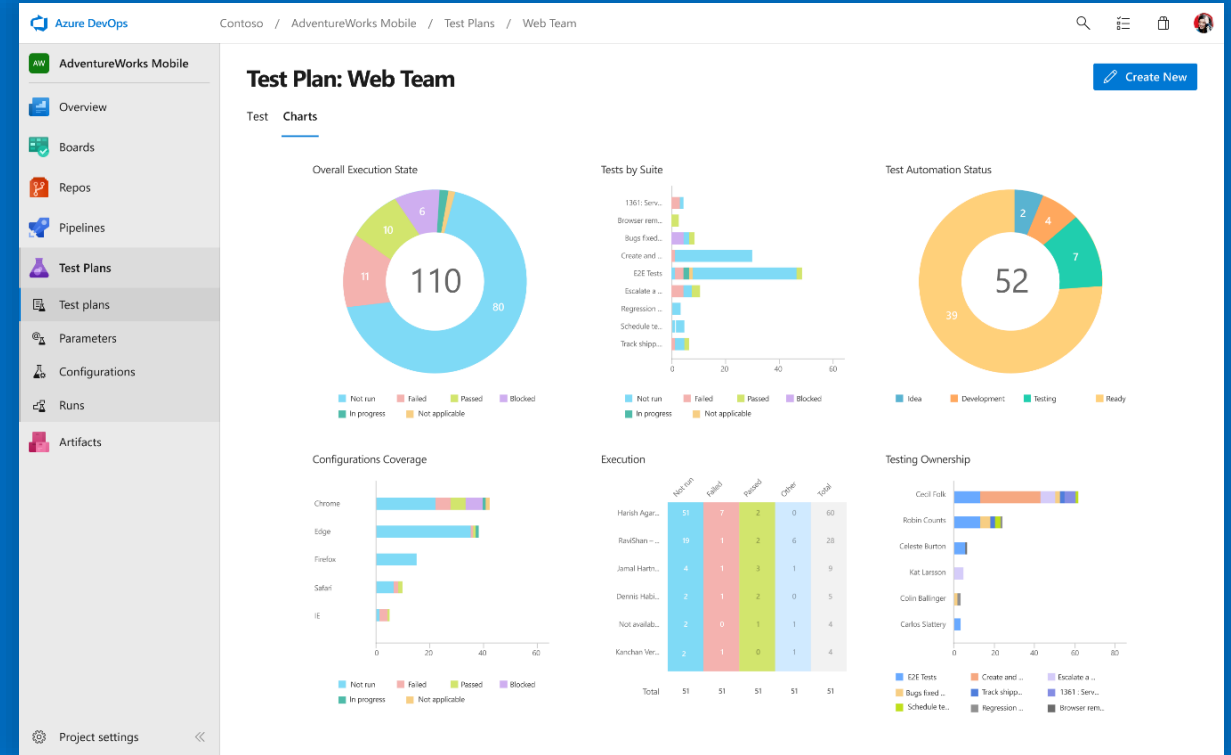
Azure Pipelines



Azure Artifacts



Azure Test Plans



Run tests and log defects from your browser

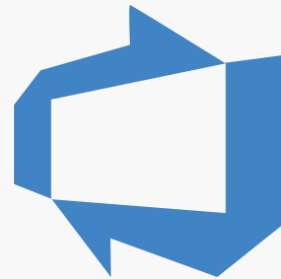
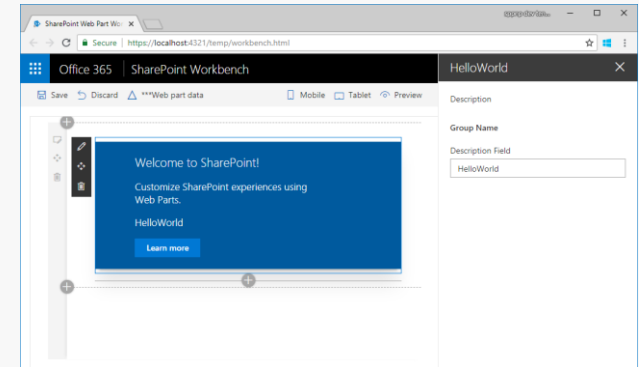
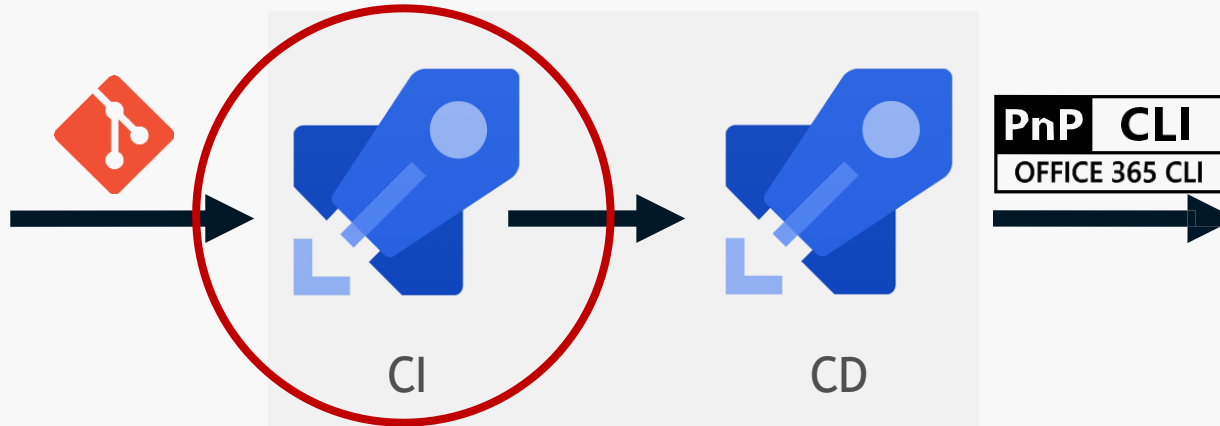
Track and assess quality throughout your lifecycle

Capture rich data for reproducibility

Create tests directly from exploratory sessions

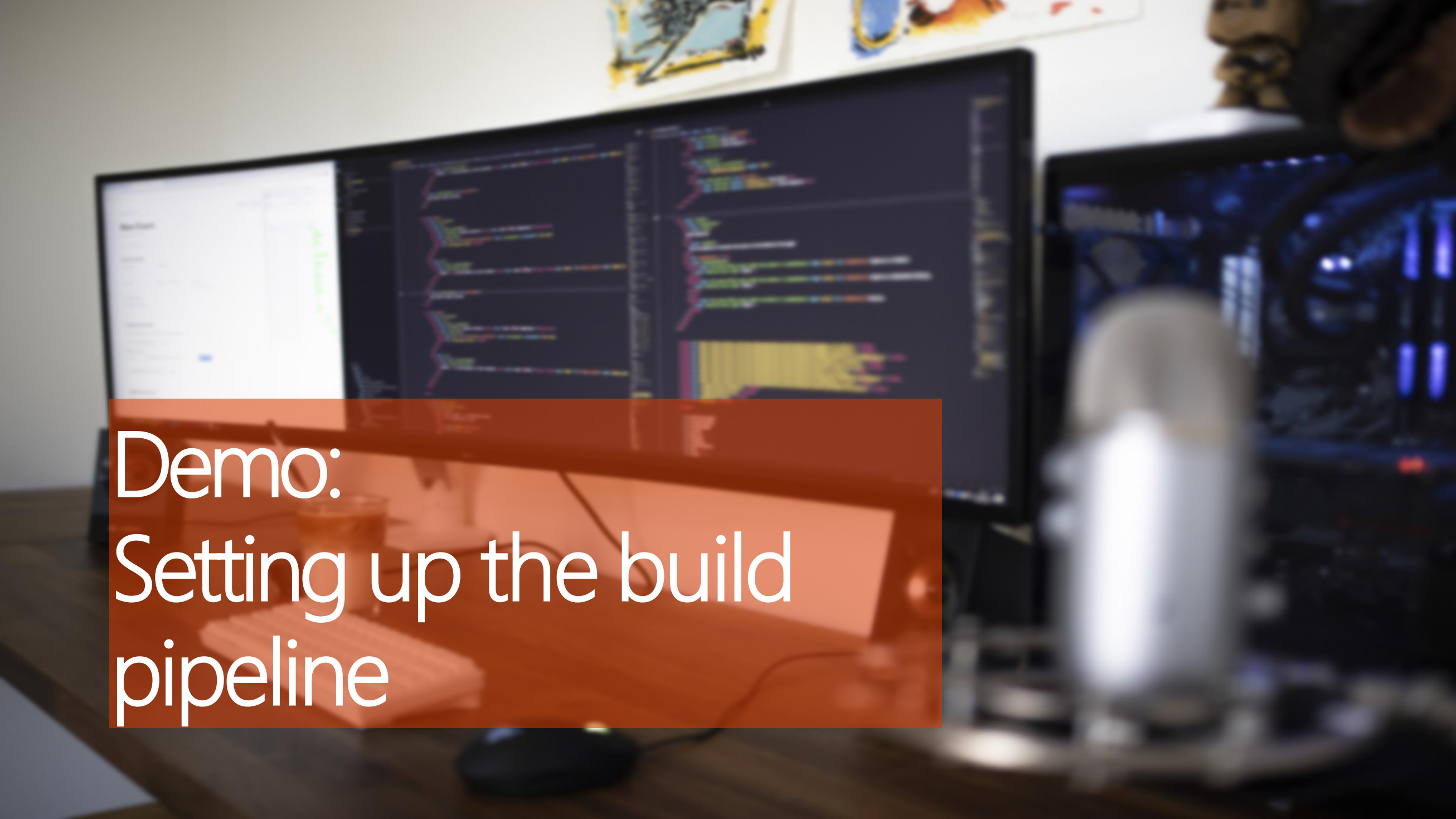
What is the process to get it automated?

```
1 import * as React from 'react';
2 import styles from './HelloWorld.module.scss';
3 import { HelloWorldProps } from './HelloWorldProps';
4 import { escape } from '@microsoft/sp-lodash-subset';
5
6 export default class HelloWorld extends React.Component<HelloWorldProps, {}> {
7   public render(): React.ReactElement<HelloWorldProps> {
8     return (
9       <div className={styles.helloworld}>
10         <div className={styles.container}>
11           <div className={styles.row}>
12             <div className={styles.column}>
13               <span className={styles.title}>Welcome to SharePoint!</span>
14               <p className={styles.subtitle}>Customize SharePoint experiences using Web Parts.</p>
15               <p className={styles.description}>{escape(this.props.description)}</p>
16               <a href="https://aka.ms/spis" className={styles.button}>
17                 <span className={styles.label}>Learn more</span>
18               </a>
19             </div>
20           </div>
21         </div>
22       </div>
23     );
24   }
25 }
```



Continuous Integration → build process





Demo: Setting up the build pipeline

Be one of the cool kids, start using YAML

- Pipelines as code
- Automated from your repo
- Easier to go back to earlier builds
- Defined in: azure-pipelines.yml
- Set secret variables via the web UI

```
1 parameters:
2   name: ''
3 jobs:
4   - job: ${ parameters.name }
5     pool:
6       vmImage: 'ubuntu-latest'
7     demands:
8       - npm
9       - node.js
10      - java
11    variables:
12      npm_config_cache: $(Pipeline.Workspace)/.npm
13
14    steps:
15      - checkout: self
16
17      - task: NodeTool@0
18        displayName: 'Use Node 10.x'
19        inputs:
20          versionSpec: 10.x
21          checkLatest: true
22
23      - task: CacheBeta@1
24        inputs:
25          key: npm | $(Agent.OS) | package-lock.json
26          path: $(npm_config_cache)
27          cacheHitVar: CACHE_RESTORED
28      - script: npm ci
29        displayName: 'npm ci'
30
31      - task: Gulp@0
32        displayName: 'Bundle project'
33        inputs:
34          targets: bundle
35          arguments: '--ship'
```

Task groups: Reuse tasks in other pipelines / projects

The screenshot shows the 'Task groups' interface in Azure DevOps, specifically for a group named 'SPFx build tasks'. The breadcrumb navigation at the top indicates the path 'Task groups > SPFx build tasks'. Below this, there are tabs for 'Tasks', 'History', and 'References', with 'Tasks' being the active tab. To the right of the tabs are buttons for 'Refresh', 'Save', 'Export', and a menu icon. The main content area displays a list of tasks under the heading 'SPFx build tasks' with a version indicator 'Version 1.*' and a plus icon for adding new tasks. The tasks listed are:

- Use Node 8.x** (Node Tool Installer)
- npm install** (npm)
- Bundle solution** (Gulp)
- Package solution** (Gulp)
- Run SonarQube Config** (npm)
- Command Line Script** (Command Line)
- Run SonarQube Analysis** (npm)
- Archive files** (Archive Files)
- Publish Artifacts: \$(Build.BuildId)-\$(Build.SourceBranchNa...** (Publish Build Artifacts)

Task groups: Reuse tasks in other pipelines / projects

The screenshot displays the Azure Pipelines configuration interface. At the top, there is a navigation bar with tabs: **Tasks**, Variables, Triggers, Options, Retention, and History. To the right of these tabs are buttons for 'Save & queue' and 'Dis' (likely 'Discard'). Below the navigation bar, the main content area is divided into sections. The first section is a light blue header labeled 'Pipeline' with the subtitle 'Build pipeline' and a three-dot menu icon. The second section is a light gray box labeled 'Get sources' with the subtitle 'valo-templating' and a 'dev' branch indicator. The third section is a white box labeled 'Phase 1' with the subtitle 'Run on agent' and a plus icon. The fourth section is a white box labeled 'Task group: SPFx build tasks' with the subtitle 'SPFx build tasks' and a blue cube icon.

Tasks Variables Triggers Options Retention History | Save & queue ▾ Dis

Pipeline
Build pipeline

Get sources
valo-templating dev

Phase 1
Run on agent

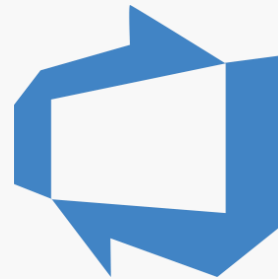
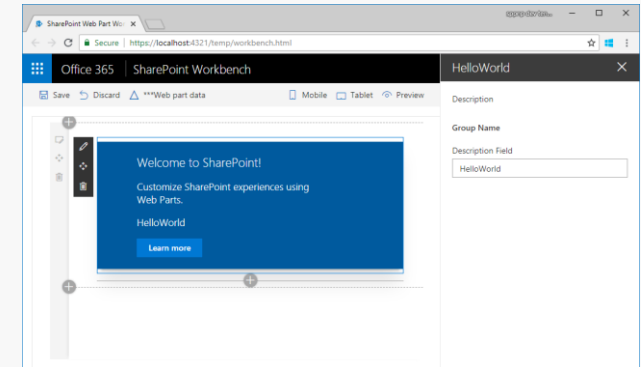
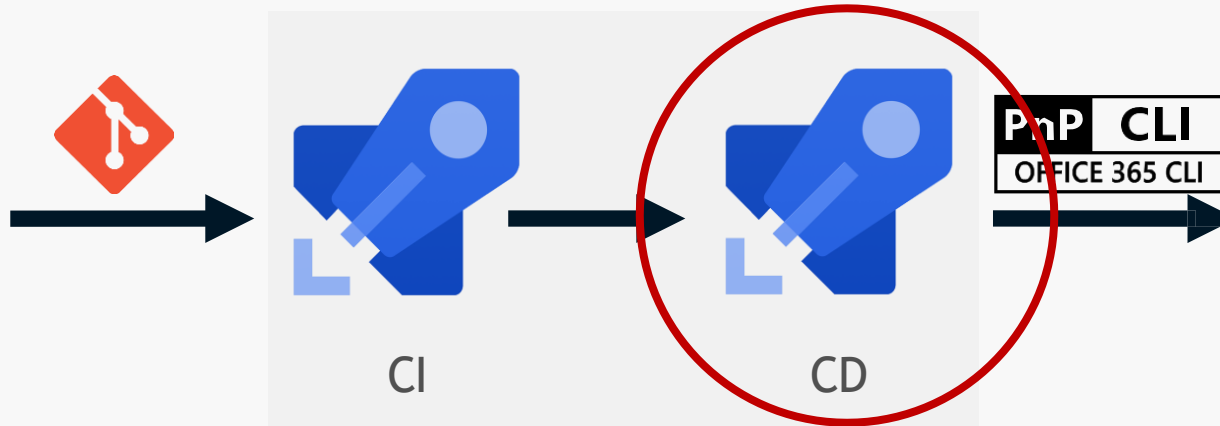
Task group: SPFx build tasks
SPFx build tasks



Demo: Task groups

Next is the release to SharePoint

```
1 import * as React from 'react';
2 import styles from './HelloWorld.module.scss';
3 import { HelloWorldProps } from './HelloWorldProps';
4 import { escape } from '@microsoft/sp-lodash-subset';
5
6 export default class HelloWorld extends React.Component<HelloWorldProps, {}> {
7   public render(): React.ReactElement<HelloWorldProps> {
8     return (
9       <div className={styles.helloWorld}>
10        <div className={styles.container}>
11          <div className={styles.row}>
12            <div className={styles.column}>
13              <span className={styles.title}>Welcome to SharePoint!</span>
14              <p className={styles.subtitle}>Customize SharePoint experiences using Web Parts.</p>
15              <p className={styles.description}>{escape(this.props.description)}</p>
16              <a href="https://aka.ms/spfx" className={styles.button}>
17                <span className={styles.label}>Learn more</span>
18              </a>
19            </div>
20          </div>
21        </div>
22      </div>
23    );
24  }
25 }
```



Continuous Delivery → Release process





Demo: Setting up the release pipeline

Q&A

Serdar Ketenci - @serdarktnci
ketenci.serdar@hotmail.com
<https://www.serdarketenci.com>