# Storybook for Angular

Table of Contents

[Storybook for Angular 1](#_Toc120611415)

[Introduction 3](#_Toc120611416)

[What is a story? 3](#_Toc120611417)

[Storybook for angular 3](#_Toc120611418)

[Installing Storybook in your angular project 3](#_Toc120611419)

[Installing angular in an empty project 5](#_Toc120611420)

[Setting up storybook 6](#_Toc120611421)

[Configuration 6](#_Toc120611422)

[Navigating storybook 7](#_Toc120611423)

[Views in storybook 8](#_Toc120611424)

[Canvas 8](#_Toc120611425)

[Docs 8](#_Toc120611426)

[Features in Canvas 8](#_Toc120611427)

[Change background 9](#_Toc120611428)

[Grid 9](#_Toc120611429)

[Measure 9](#_Toc120611431)

[Preview Size 10](#_Toc120611432)

[Outline 10](#_Toc120611433)

[Vision Simulator 11](#_Toc120611434)

[Controls 12](#_Toc120611435)

[Actions 12](#_Toc120611436)

[Accessibility 13](#_Toc120611437)

[Conclusion 14](#_Toc120611438)

Gomathinayagam A



# 

# Introduction

Storybook is a UI component development environment that allows developers to create and test UI components in isolation. This enables us to work on a single module at a time. It runs alongside the application in development mode. Storybook supports Angular, React, React Native, Vue, Svelte and Ember. Storybook can also help us document and reuse UI components, hence it makes building web applications faster and more efficient.

# What is a story?

Story is a rendered state of a UI component. There can be multiple stories per component to describe different states of a component.

# Storybook for angular

Storybook is used to develop angular components. Developers can create stories for their components. Each story shows the behavior for a single component with specific states set.

## Installing Storybook in your angular project

To install storybook in an existing angular application run the following command

**npx storybook init**

and update angular.json to include Storybook’s custom builder



This command will only work on non-empty projects, it will look into the project’s dependencies during

the installation process and provide the best configuration available.

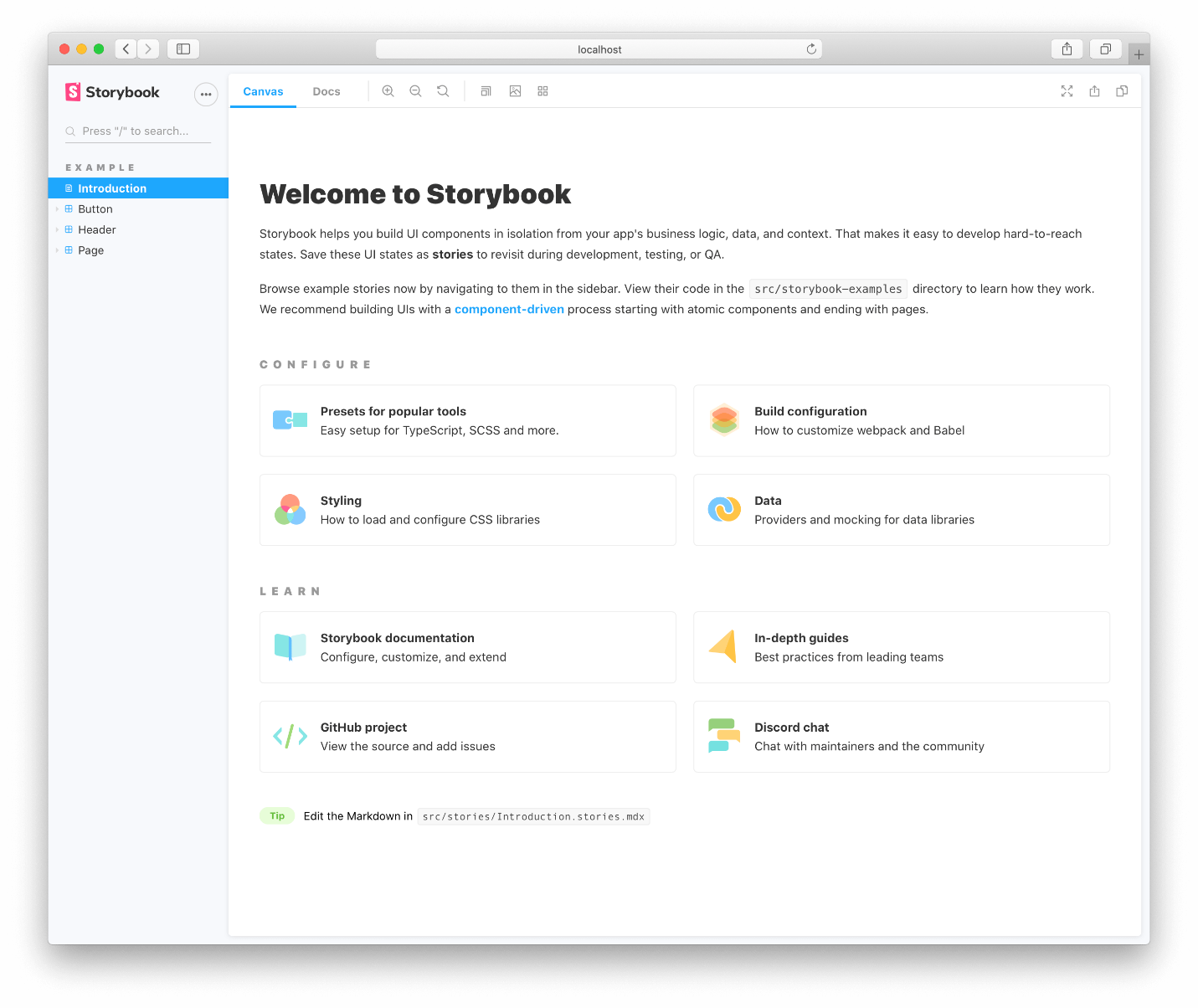
This command will make the following changes to your local environment:

* Install the required dependencies.
* Setup the necessary scripts to run and build Storybook.
* Add the default Storybook configuration.
* Add some boilerplate stories to get you started.
* Set up telemetry to help us improve Storybook

You can then run

**npm run storybook**

command to run storybook, which will open the welcome screen automatically.



# 

## Installing angular in an empty project

First install storybook in the project directory with the following command

**npm i -g @storybook/cli @angular/cli**

then run the init command

**sb init**

this will create all files required for storybook. Now run the storybook with the run command.

**npm run storybook**

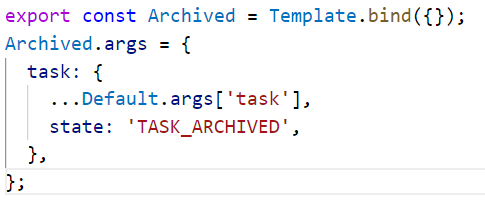
this will take you to the welcome screen

# setting up storybook

Create a story file for the component to be tested. If the component file name is

**task.component.ts** the story file should be **task.stories.ts** in the same directory.

In the stories file the states of the component are defined



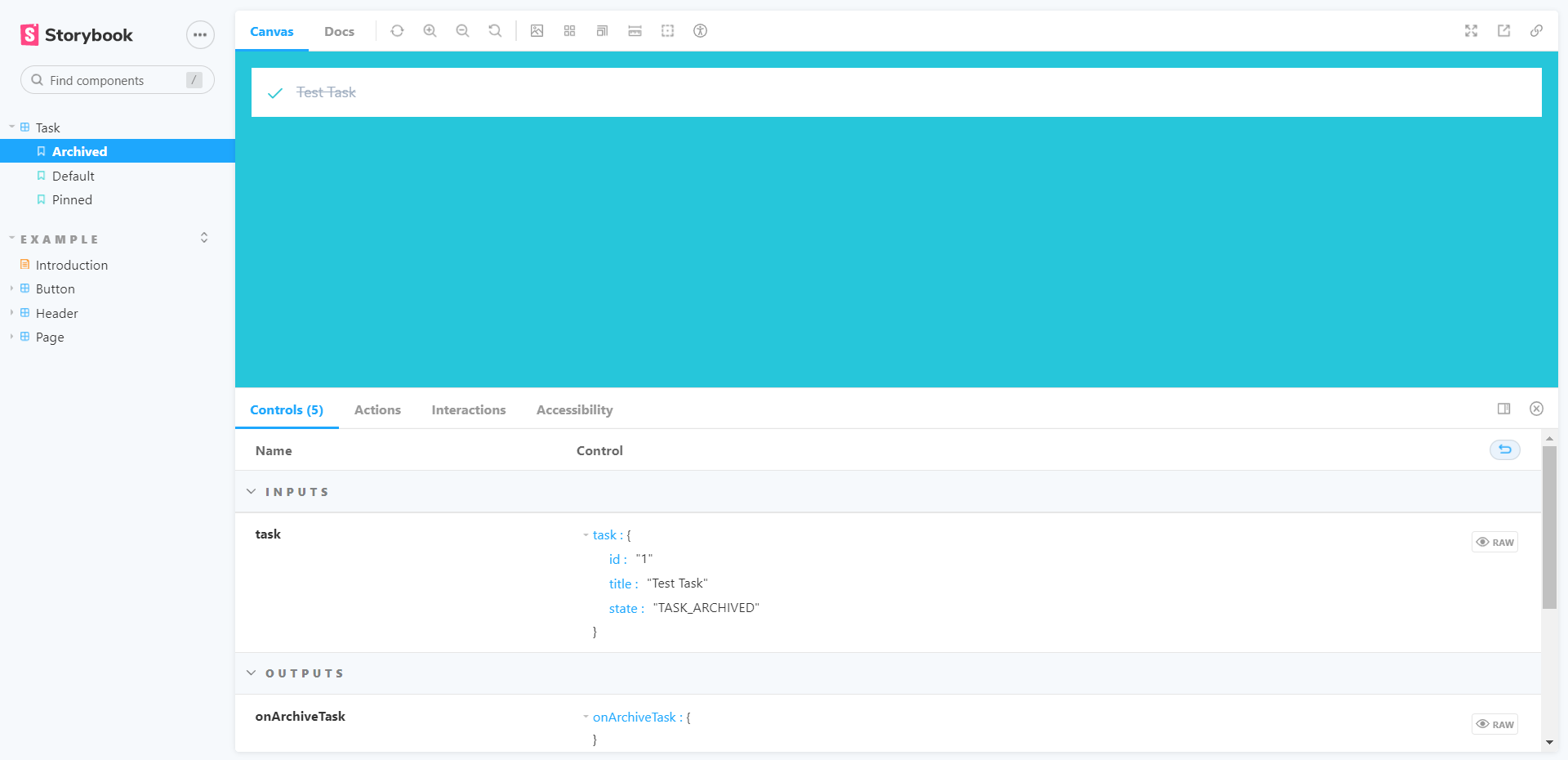
In the above sample code, a state named ‘TASK\_ARCHIVED’ is created. Similarly, many states can be created for a single component. There are two basic levels of organization in Storybook: the component and its child stories. Stories are like permutations of a component. You can have as many stories per component as you need.

## Configuration



In the source directory after installing storybook you will notice that a new folder **‘.stories’** is created inside which, there is ‘**main.js**’. These are the configurations for storybook.

# navigating storybook



once you run the command **npm run storybook** the storybook UI appears, here you can see the components listed on a menu to the left. If a component is selected all states of that component is listed

Graphical user interface, application

Description automatically generated

in this image,

* Task is the component
* Archived, Default and Pinned are three states of the component.

# Views in storybook

Graphical user interface, text, application

Description automatically generated

## Canvas

After selecting a component and a state, you must select canvas to view the component in rendered form. This shows that html version of the component, how it would be rendered in your application.

## Docs

If the docs view is selected after selecting the component, it shows a detailed description of the component, its states, the methods defined inside the component, etc.…

# features in Canvas

Graphical user interface

Description automatically generated with medium confidence

While canvas view is selected there are small icons that appear on the top of the window. These options can be used to test the UI component.

## Change background

Graphical user interface, text, application

Description automatically generated

This option lets to change the background of the component to check the visibility of the component is different backgrounds.

## Grid

## Table, timeline Description automatically generated

This lets you to observe your component with a grid background, to test the orientation of the component.

## Measure

Graphical user interface, text, application

Description automatically generated

This shows the dimensions of the component rendered, on hovering over a component it will also show the width of the padding.

## Preview Size

A picture containing graphical user interface

Description automatically generated

This option allows us to view our component is different screen sizes like tablet, phone and desktop.

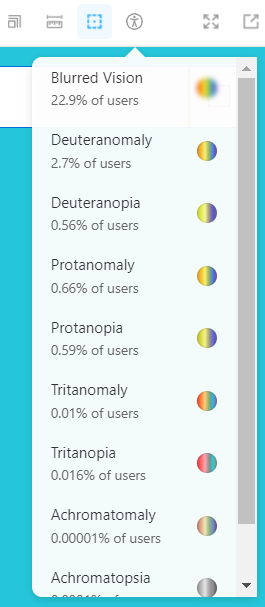
## Outline

Text

Description automatically generated with medium confidence

This draws an outline of all elements present in the component.

## Vision Simulator



This is used to test components for accessibility features, to check if the component is clearly visible for people if impaired vison. These options simulate those impairments, so developers can change the component’s design accordingly.

## Controls

Graphical user interface, application

Description automatically generated

This shows all information about the component’s logic. The inputs the component gets, values that are emitted by the component, and methods defined in the component.

## Actions

Graphical user interface, application

Description automatically generated

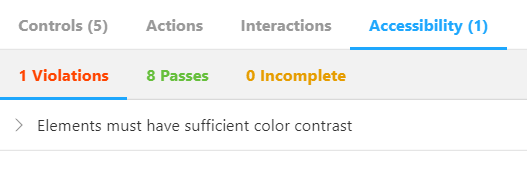
Here method calls and other actions that happens in the components are observed.

## Accessibility

Graphical user interface, text, application, email

Description automatically generated

This shows how well the component has met the accessibility tests. If there are any violations listed, they can be changed to pass the tests. For instance, if the background color of the component is the same as the foreground color that is a violation.



# Conclusion

In this document we have gone through What storybook is, and how it can be used to test and develop UI components in isolation. We have learnt about the various features of Storybook like Accessibility, Actions, Vision Simulator, Preview size, Measure, Grid and other features in canvas view. Also we have learnt about the commands and procedure to install and configure storybook in your existing project or an empty project.