

Bellissima

A fully new modern front end to Backoffice, replacing Angular JS

Thanks!

A big shout-out to Kevin Jump

<https://dev.to/kevinjump/series>

New Framework

- Everything in the new backoffice mainly comprises of:
 - Manifests
 - Web Components
 - Contexts
- Preferred tools
 - Vite
 - Lit
 - Typescript

Diving straight in

Building a property Editor

Creating a Custom Property Editor

1. Create a Package Manifest, Similar to before, but with a `propertyEditorSchema` to define the configuration settings and a `propertyEditorUi` which defines how the users sees when editing content (Controller and View)
2. Create a `DataEditor` C# Class
3. Optionally create a C# Configuration Editor (to define how the data is serialised in / out of the DB)
4. Optionally create a C# Value Editor for things like server side validation

New Concepts

- Package Entry Points
- Contexts & Resources
- Sections, sidebars and menus
- Workspaces
- Workspace views (Content Apps)
- Header apps and modal dialogs
- Actions
- dialogs

Entry Points

```
{
  "$schema": "../umbraco-package-schema.json"
  "name": "mypackage",
  "id": "mypackage",
  "version": "0.1.0",
  "allowTelemetry": true,
  "extensions": [
    {
      "name": "mypackage.entrypoint",
      "alias": "mypackage.EntryPoint",
      "type": "entryPoint",
      "js": "/app-plugins/mypackage/assets.js"
    }
  ]
}
```

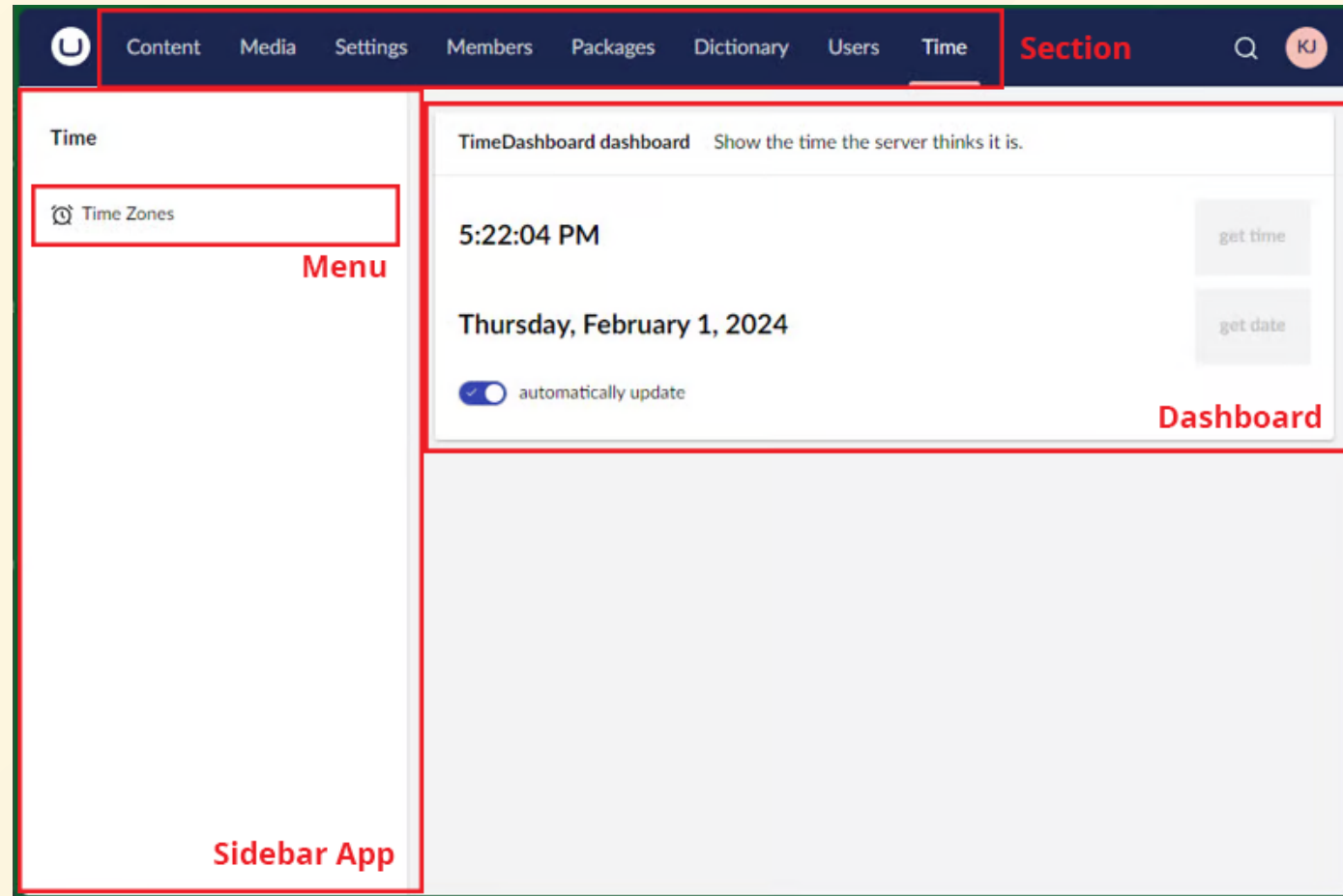
```
const TimeManifest:
  Array<ManifestGlobalContext> = [
    {
      type: 'globalContext',
      alias: 'time.context',
      name: 'Time context',
      js: () => import('./time.context')
    }
  ]

export const manifests = [
  ...TimeManifest
];
```

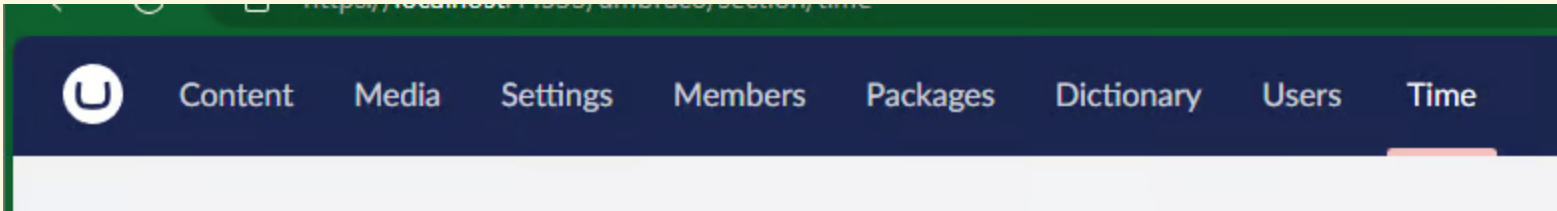
Contexts & Resources

- **Context** is like a service, it's the method you want to get things in your dashboard, etc. it also stores some values, which you can 'observe' for changes that might happen elsewhere.
- **Repositories** handle the first part of getting things, they abstract away how the data is stored in your app
- **Resources** are the things that actually go fetch the data, in the case of the back office, these are the bits doing the http requests.

Sections, Sidebars and Menus



1. Sections



```
const sectionManifest : ManifestSection = {  
  type: 'section',  
  alias: 'time.section',  
  name: 'time section',  
  weight: 10,  
  meta: {  
    label: 'Time',  
    pathname: 'time'  
  }  
}
```

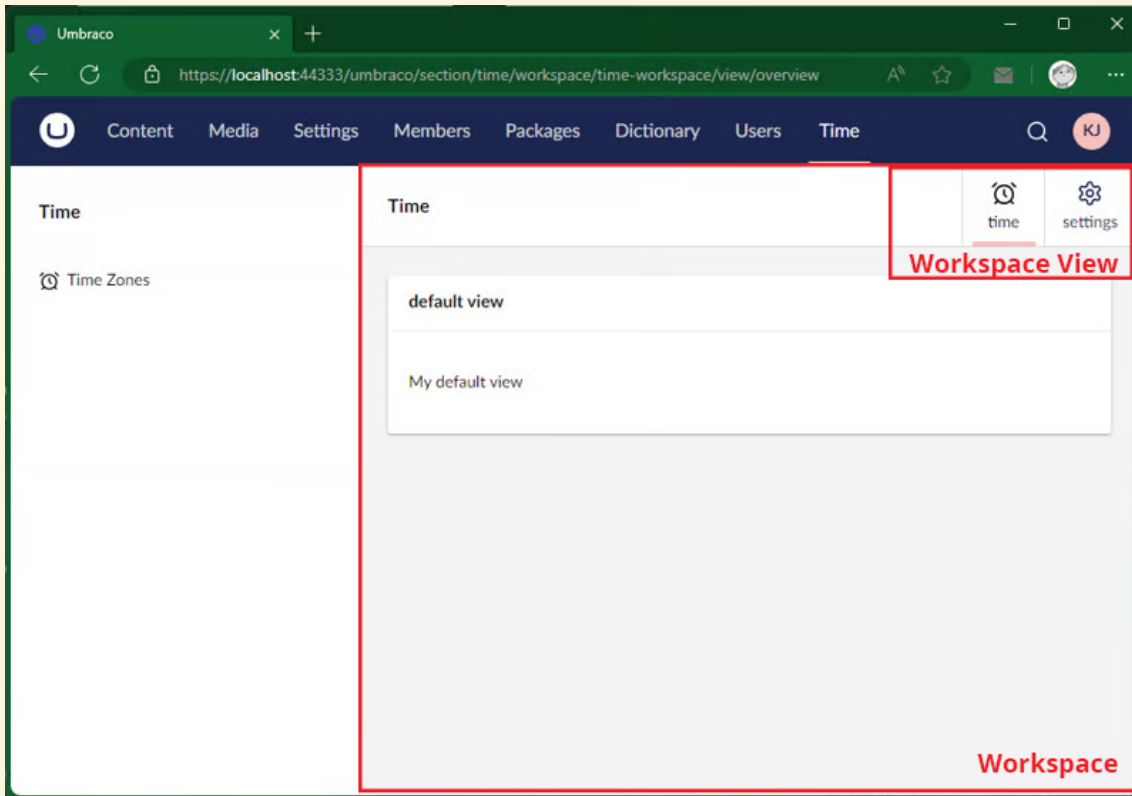
2. Sidebar App

```
const sidebarAppManifest : ManifestSectionSidebarApp = {  
  type: 'sectionSidebarApp',  
  kind: 'menuWithEntityActions',  
  alias: 'time.sidebar.app',  
  name: 'Sidebar app',  
  meta: {  
    label: "Time",  
    menu: "time.menu"  
  },  
  conditions: [  
    {  
      alias: "Umb.Condition.SectionAlias",  
      match: "time.section"  
    }  
  ]  
};
```

3. Menu and item(s)

```
const menuManifest : ManifestMenu = {  
  type: 'menu',  
  alias: 'time.menu',  
  name: 'time sidebar menu',  
  meta: { label: 'Time' }  
}  
  
const menuItemManifest : ManifestMenuItem = {  
  type: 'menuItem',  
  alias: 'time.menu,item',  
  name: 'time menu item',  
  meta: {  
    label: 'Time Zones',  
    icon: 'icon-alarm-clock',  
    entityType: '',  
    menus: [ 'time.menu' ]  
  }  
}
```

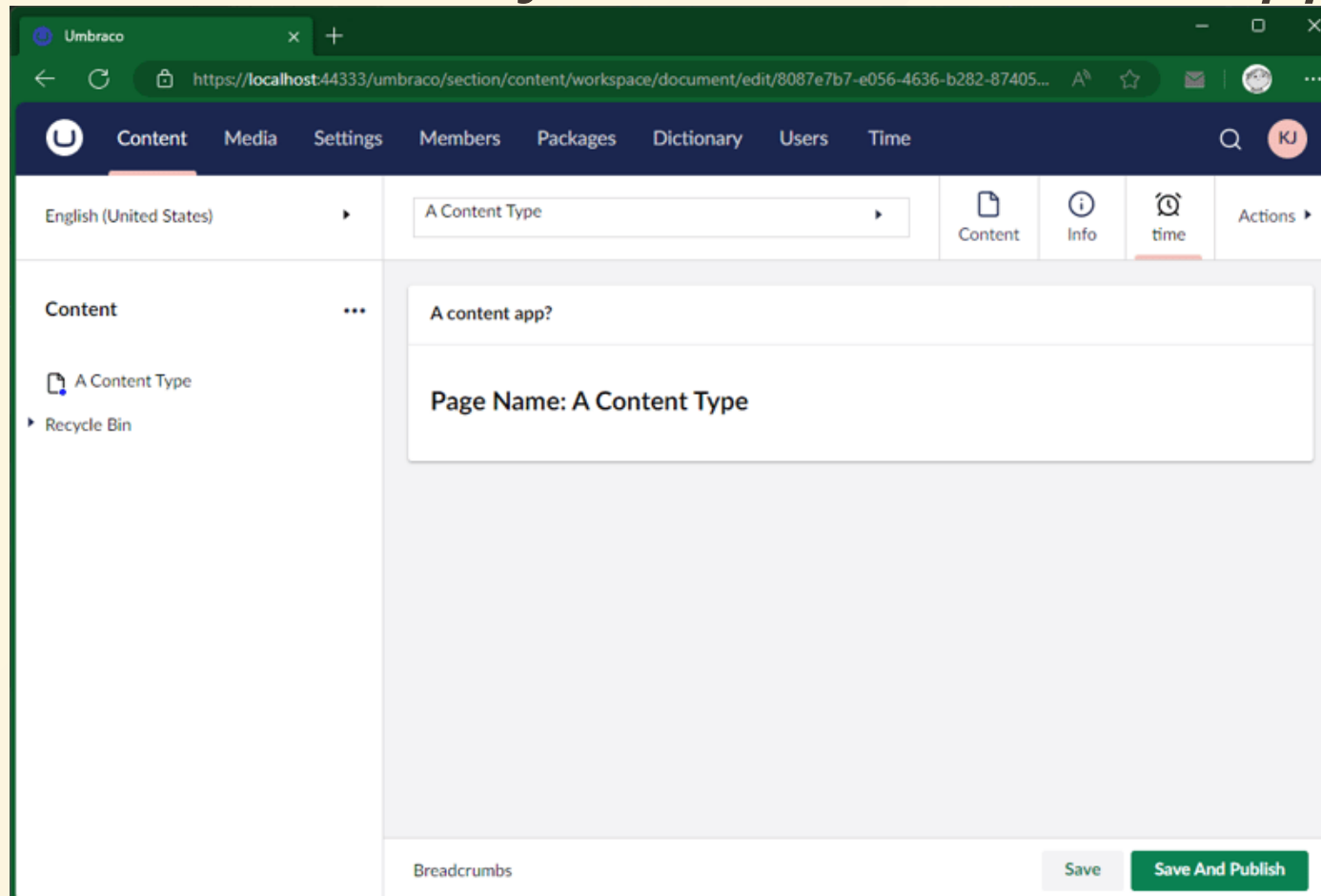
Workspaces



```
var workspaceManifest : ManifestWorkspace =  
    type: 'workspace',  
    alias: 'time.workspace',  
    name: 'time workspace',  
    element: ()=>  
        import('./workspace.element'),  
    meta: {  
        entityType: 'time-workspace'  
    }  
};
```

Workspace views

The artist formally known as "*Content apps*"



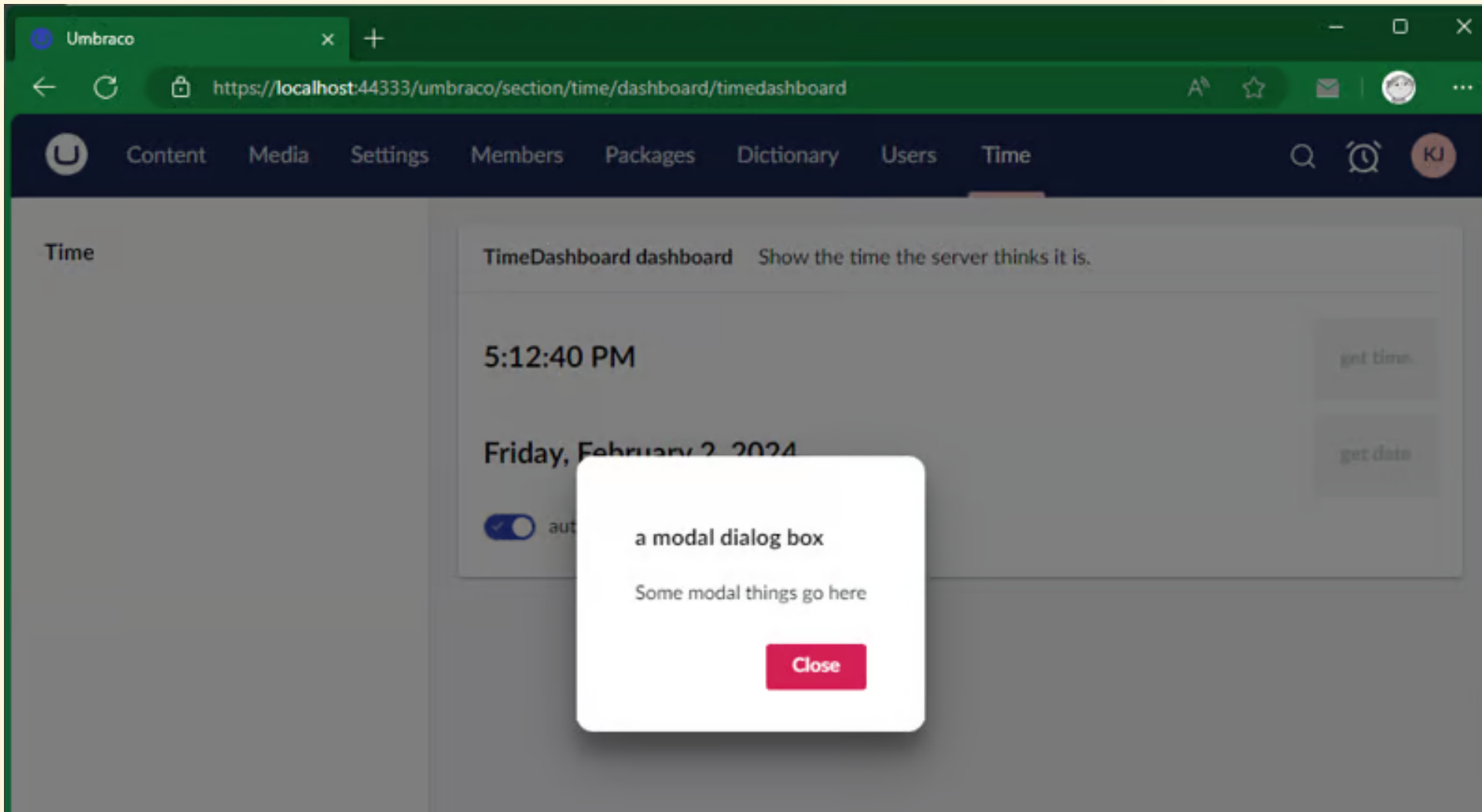
Workspace views

```
@customElement('time-document-workspace-view')
export class TimeDocumentWorkspaceElement
  extends UmbElementMixin(LitElement) {
  var pageName = "";
  constructor() {
    super();
    this.consumeContext(UMB_WORKSPACE_CONTEXT,
      (nodeContext) => {
        var variantContext = (nodeContext as UmbVariantableWorkspaceContextInterface);
        pageName = variantContext.getName();
      });
  }
  render() {
    ...
  }
}
```

Workspace views

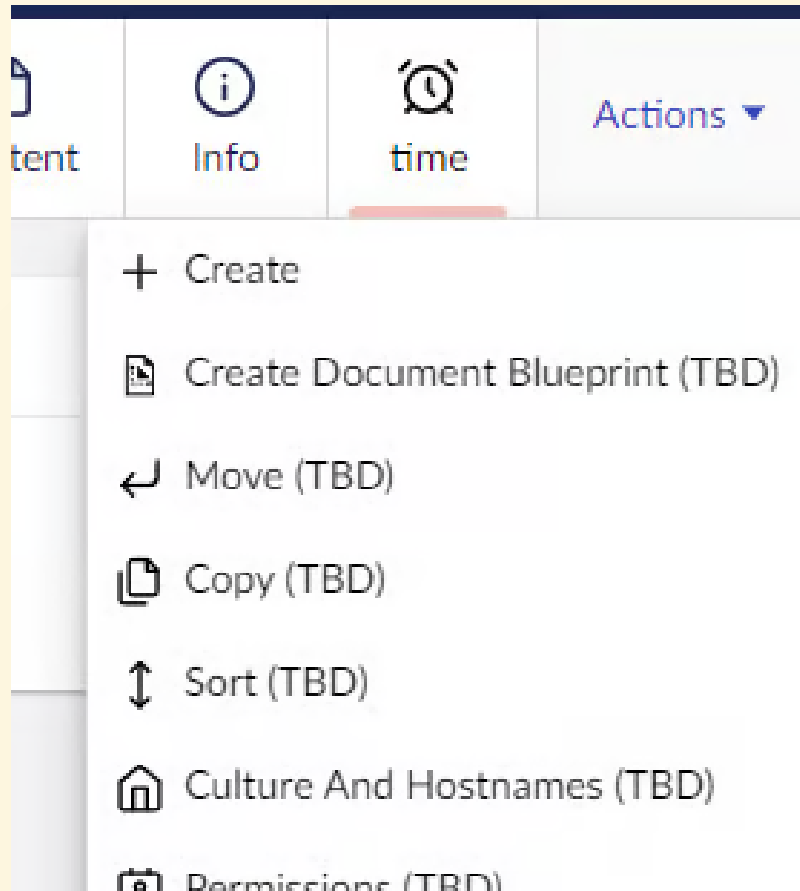
```
@customElement('time-document-workspace-view')
export class TimeDocumentWorkspaceElement
  extends UmbElementMixin(LitElement) {
  var pageName = " ";
  constructor() {
    ...
  }
  render() {
    return html`
      <uui-box headline="A content app?">{{pageName}}</uui-box>
    `;
  }
  static styles = css`
    uui-box { margin: 20px; }
  `;
}
```


Header apps and modal dialogs



Actions

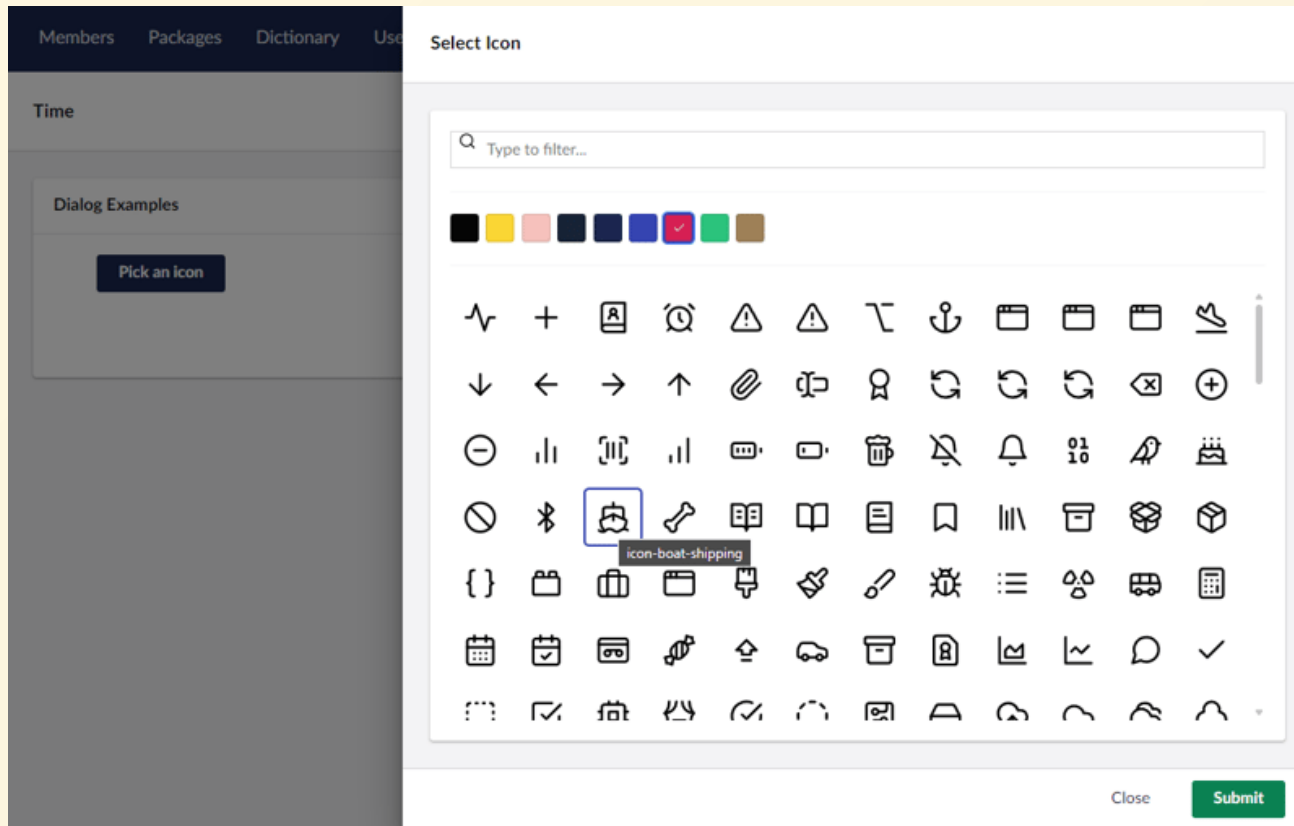
Entity Actions



Workspace Actions



Dialogs



Links

Presentation :

<https://bit.ly/bellissima-Sydney>

Early Adopters Guide to v14:

<https://dev.to/kevinjump/series>

Bellissima WIP articles:

<https://dev.to/leekelleher/umb-raco-bellissima-wip-articles-29f8>

This Presentation:

