Prepping for Obsidian

What you can do to get ready for the new UI framework

Daniel Hazelbaker

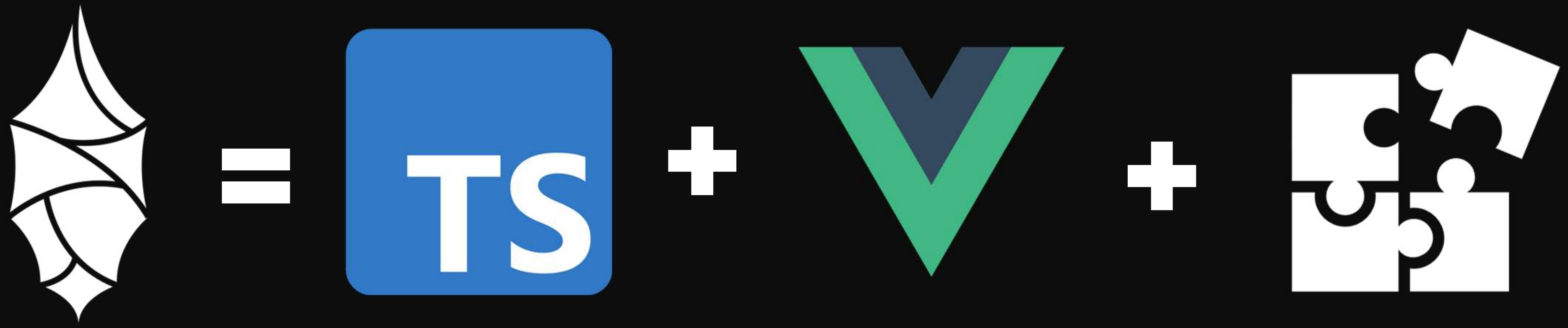
Software Developer

Working at Triumph Tech

20 Years Experience at Churches

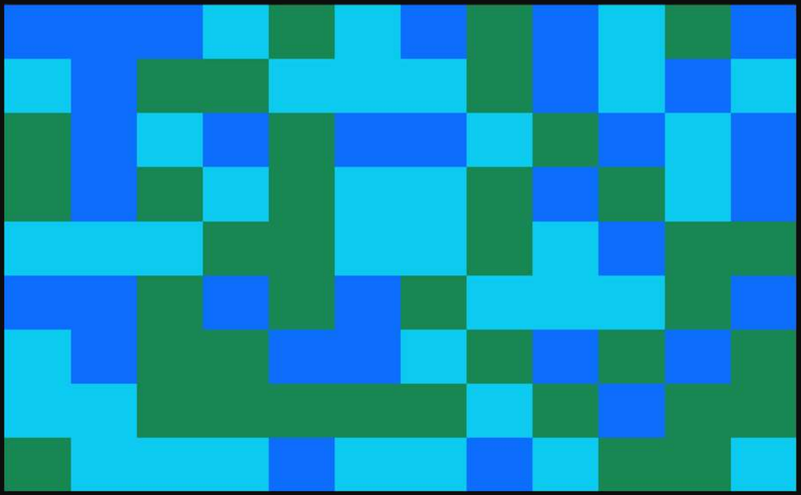
cabal95

What is Obsidian

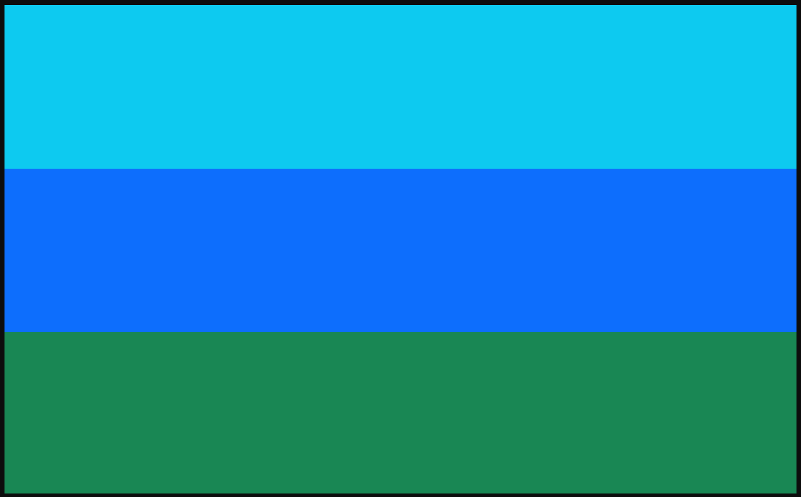


WebForms vs Obsidian

WebForms Block



Obsidian Block



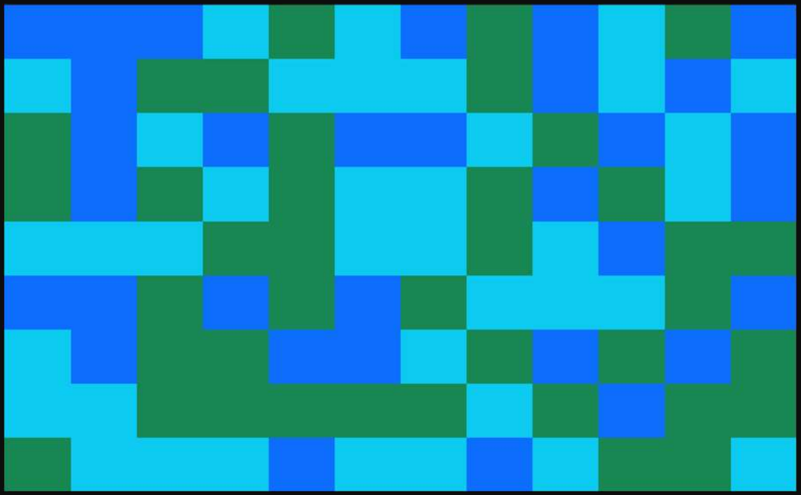
UI / Layout

Logic

Service

WebForms vs Obsidian

WebForms Block



Obsidian Block

campusDetail.obs

CampusDetail.cs

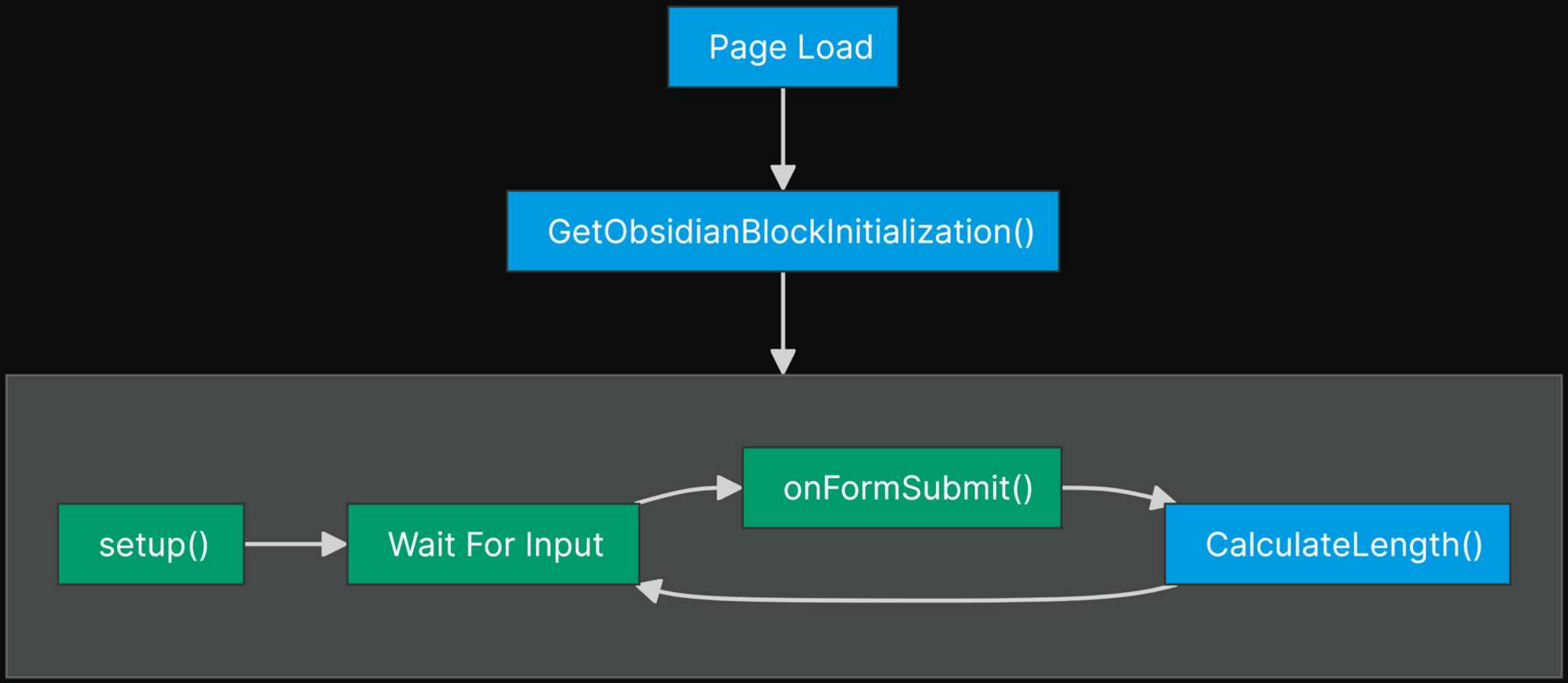
CampusService.cs

UI / Layout

Logic

Service

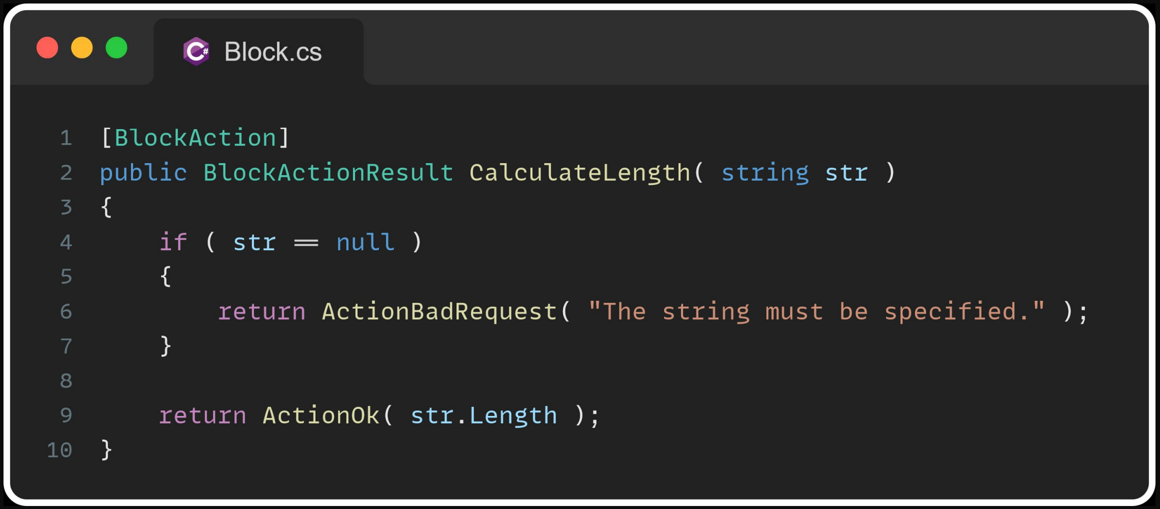
How Obsidian Blocks Work



Block Actions

A call to block action `CalculateLength` becomes a URL request.

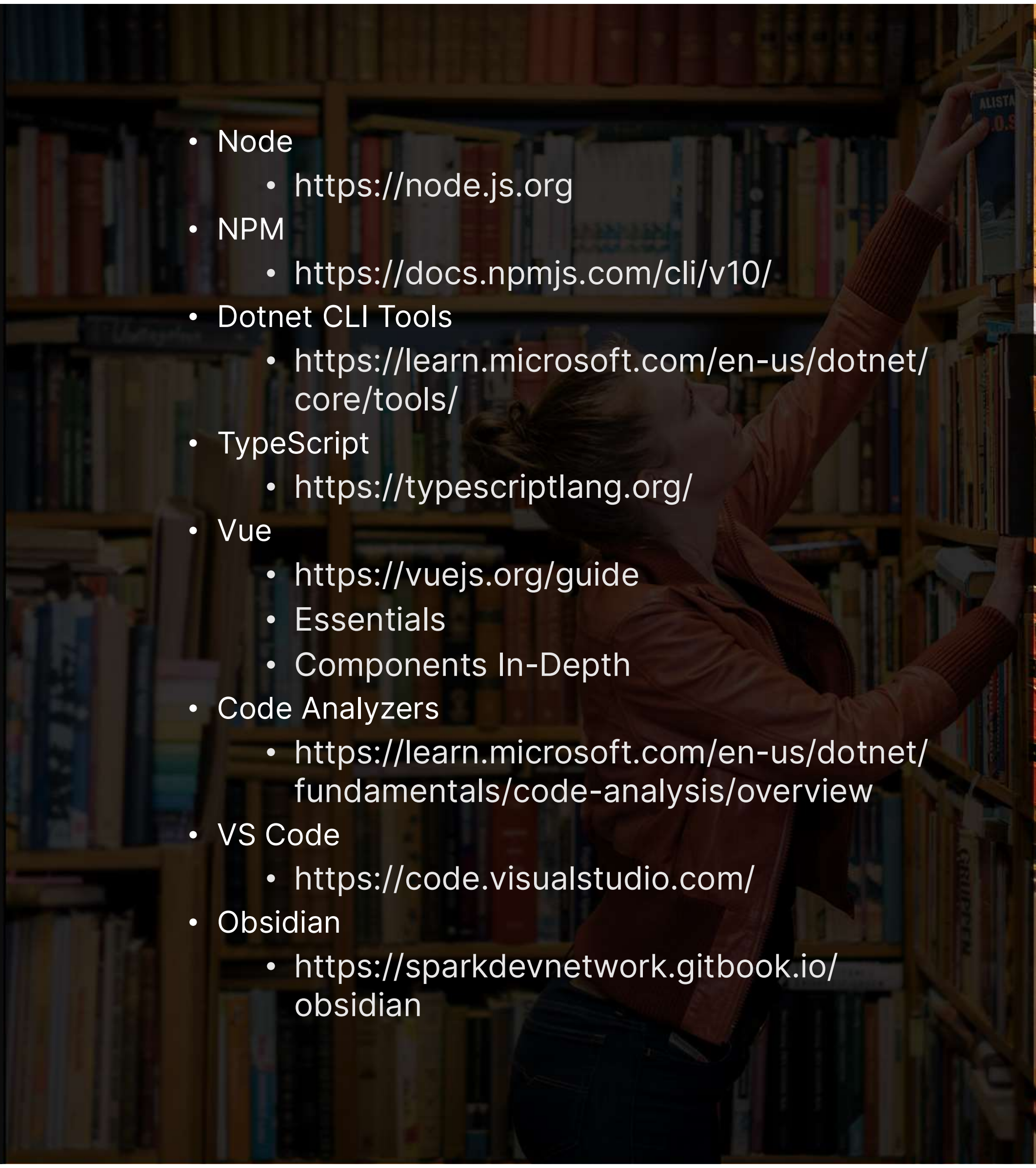
```
POST https://www.rockrms.com/api/v2/<pageguid>/<blockguid>/CalculateLength
```



Block.cs

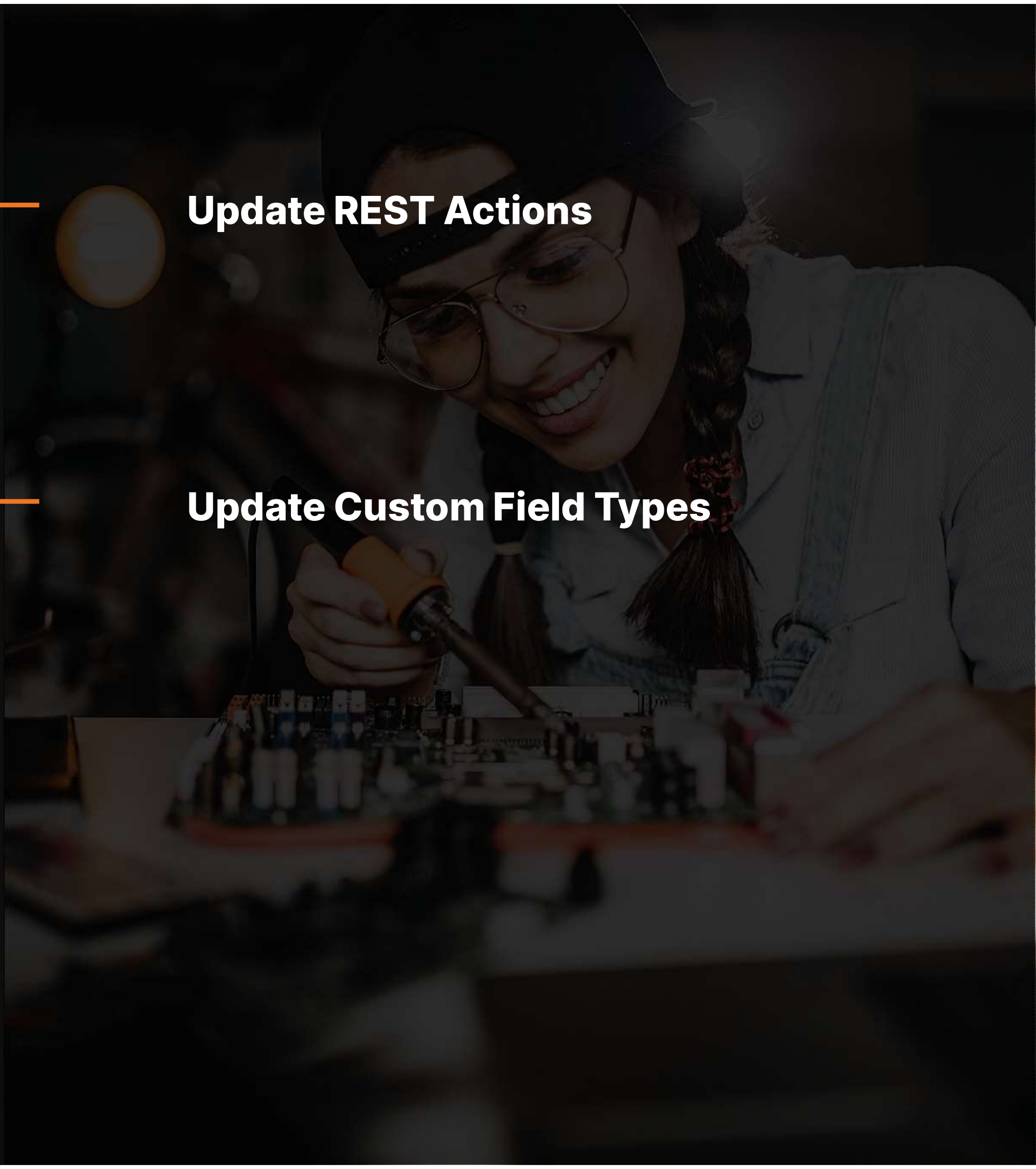
```csharp
[BlockAction]
public BlockActionResult CalculateLength( string str )
{
    if ( str == null )
    {
        return ActionBadRequest( "The string must be specified." );
    }

    return ActionOk( str.Length );
}
```

What Can You Learn Now



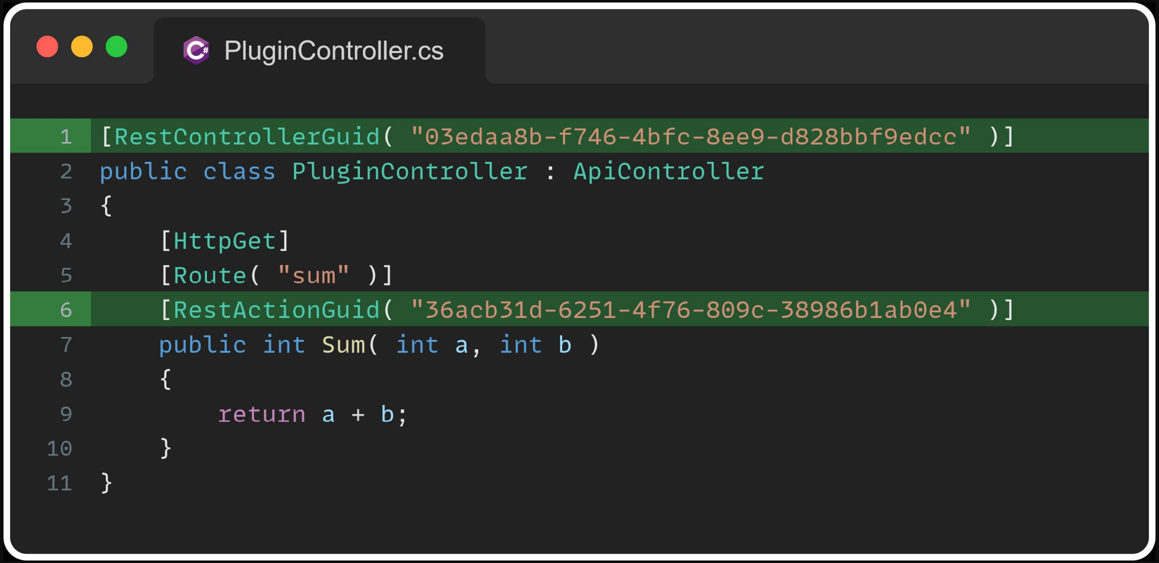
- Node
  - https://node.js.org
- NPM
  - https://docs.npmjs.com/cli/v10/
- Dotnet CLI Tools
  - https://learn.microsoft.com/en-us/dotnet/core/tools/
- TypeScript
  - https://typescriptlang.org/
- Vue
  - https://vuejs.org/guide
  - Essentials
  - Components In-Depth
- Code Analyzers
  - https://learn.microsoft.com/en-us/dotnet/fundamentals/code-analysis/overview
- VS Code
  - https://code.visualstudio.com/
- Obsidian
  - https://sparkdevnetwork.gitbook.io/obsidian

What Can Do Now



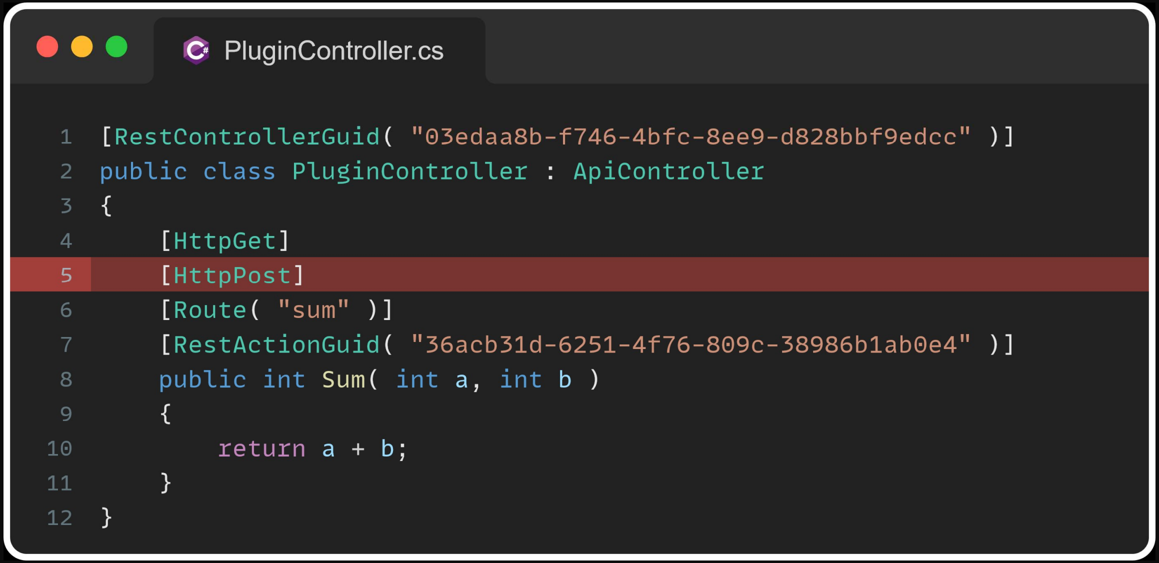
- Update REST Actions
- Update Custom Field Types

REST Actions


PluginController.cs

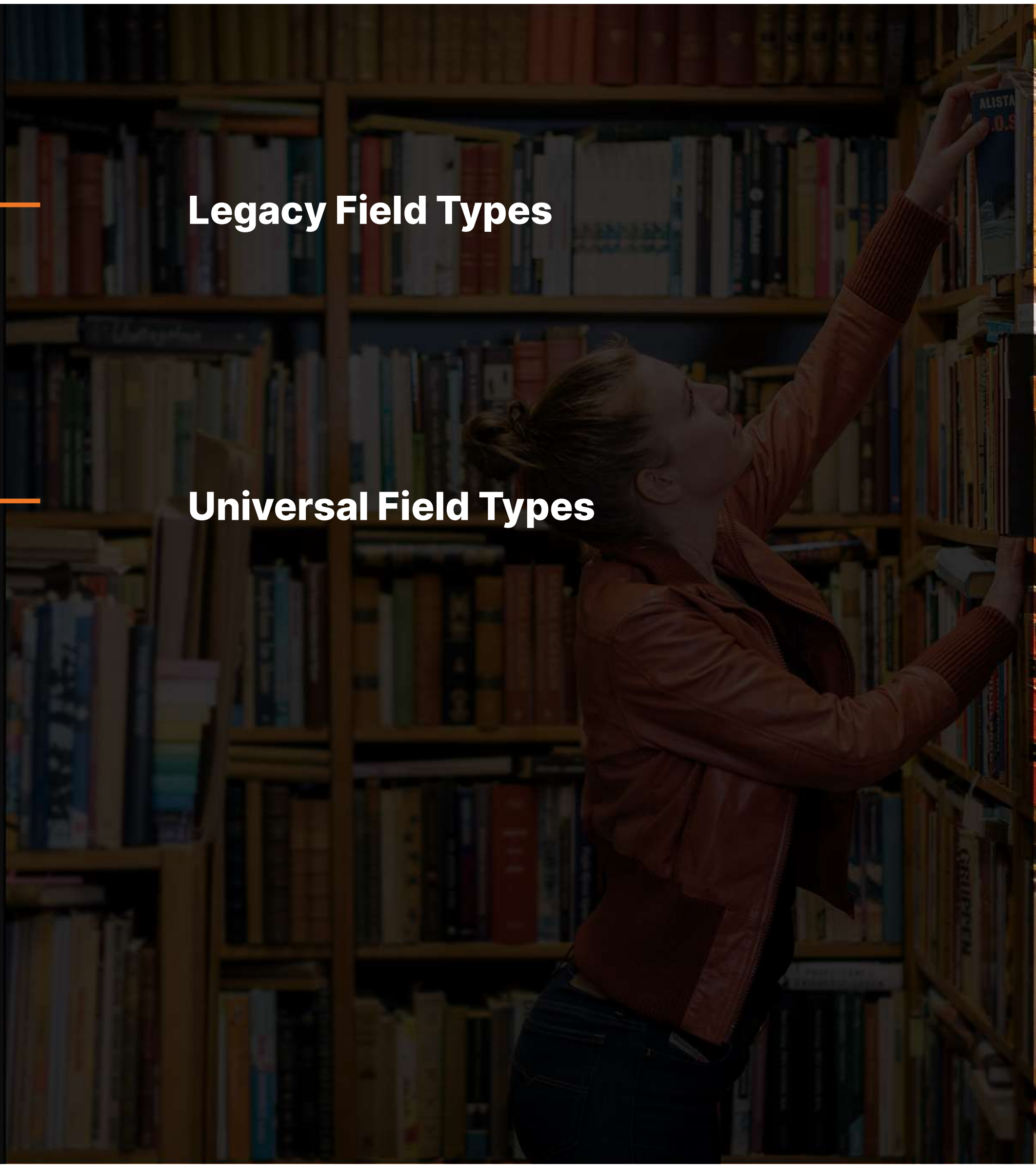
```csharp
[RestControllerGuid( "03edaa8b-f746-4bfc-8ee9-d828bbf9edcc" )]
public class PluginController : ApiController
{
    [HttpGet]
    [Route( "sum" )]
    [RestActionGuid( "36acb31d-6251-4f76-809c-38986b1ab0e4" )]
    public int Sum( int a, int b )
    {
        return a + b;
    }
}
```

REST Actions



```
[RestControllerGuid( "03edaa8b-f746-4bfc-8ee9-d828bbf9edcc" )]
public class PluginController : ApiController
{
    [HttpGet]
    [HttpPost]
    [Route( "sum" )]
    [RestActionGuid( "36acb31d-6251-4f76-809c-38986b1ab0e4" )]
    public int Sum( int a, int b )
    {
        return a + b;
    }
}
```

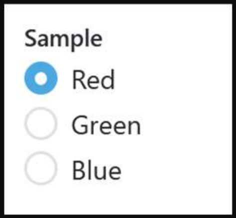
Universal Field Types

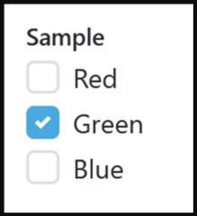


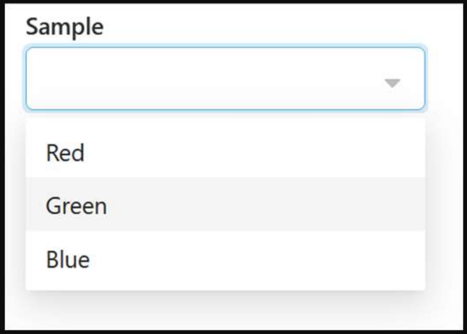
Legacy Field Types

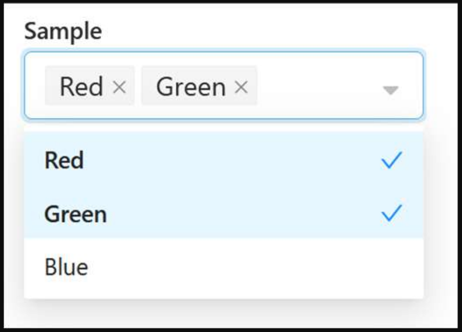
Universal Field Types

Field Types

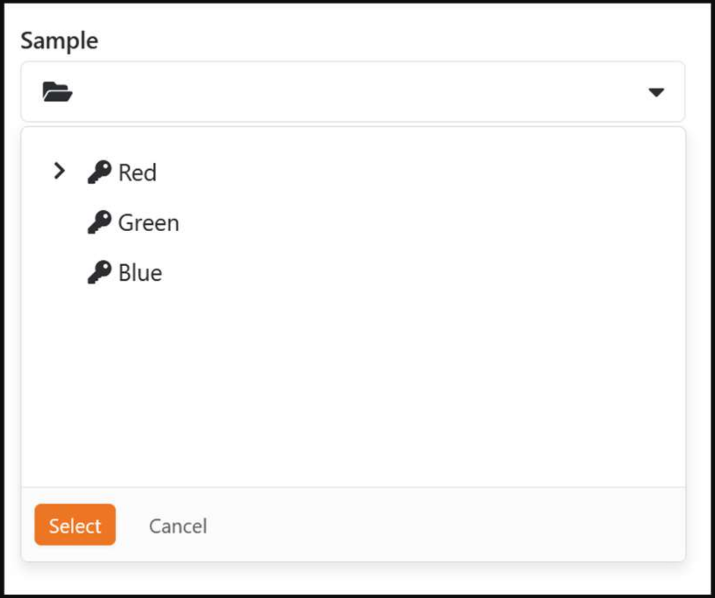

Sample
Red
Green
Blue


Sample
Red
Green
Blue


Sample
Red
Green
Blue


Sample
Red ×
Green ×
Red
Green
Blue

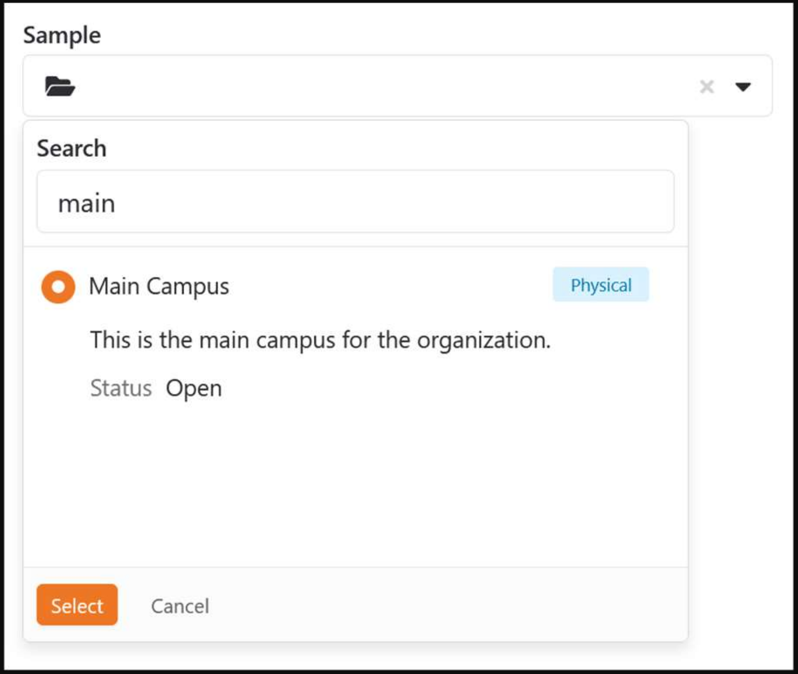
Field Types


Sample

Red

Green

Blue


Sample

Search

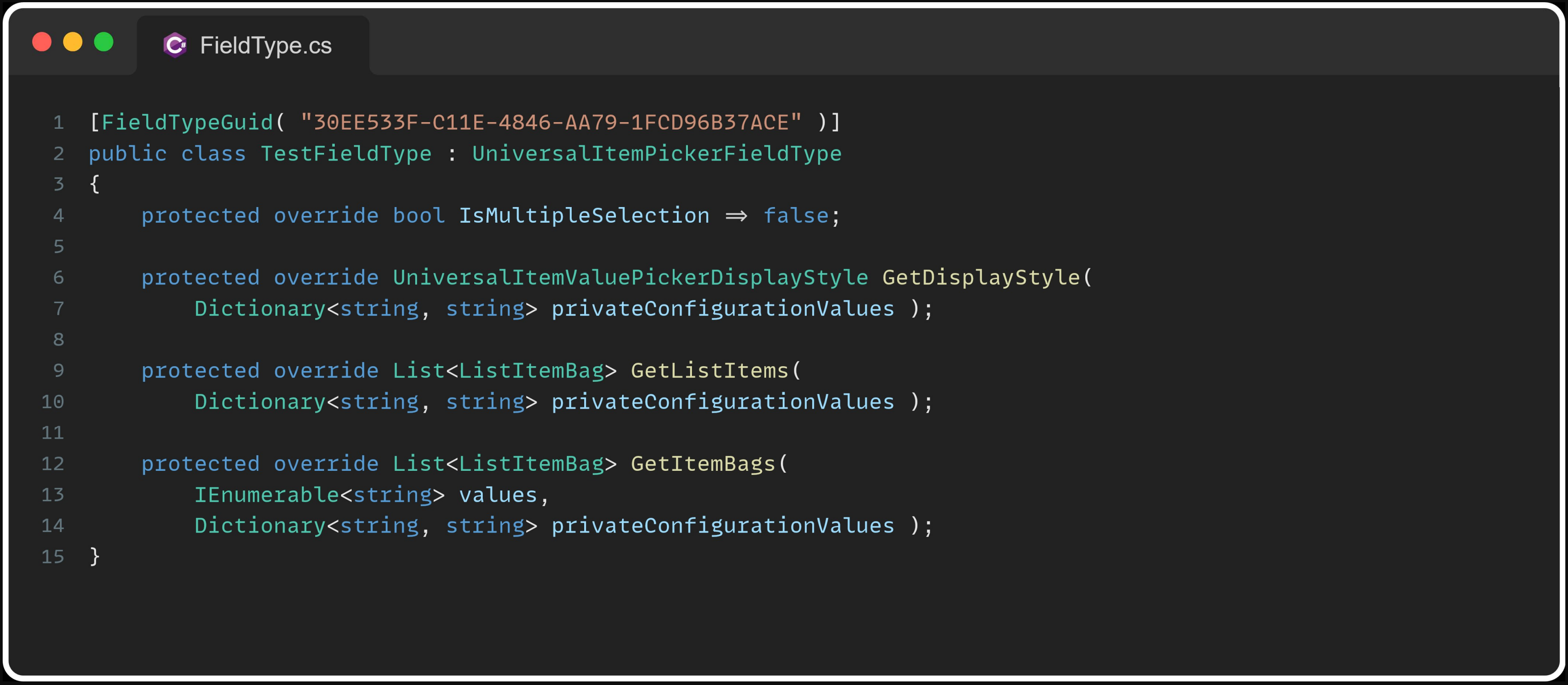
main

Main Campus

This is the main campus for the organization.

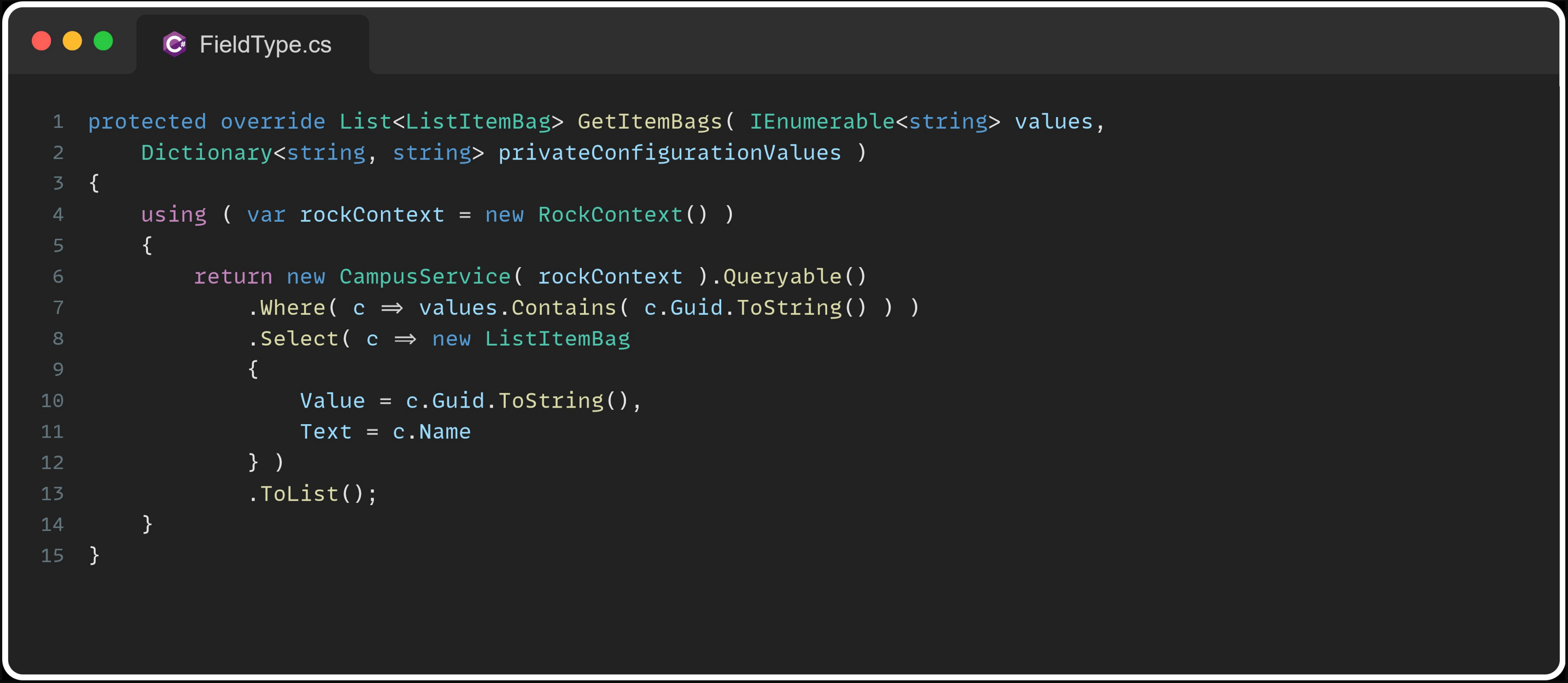
Status Open

Item Field Type



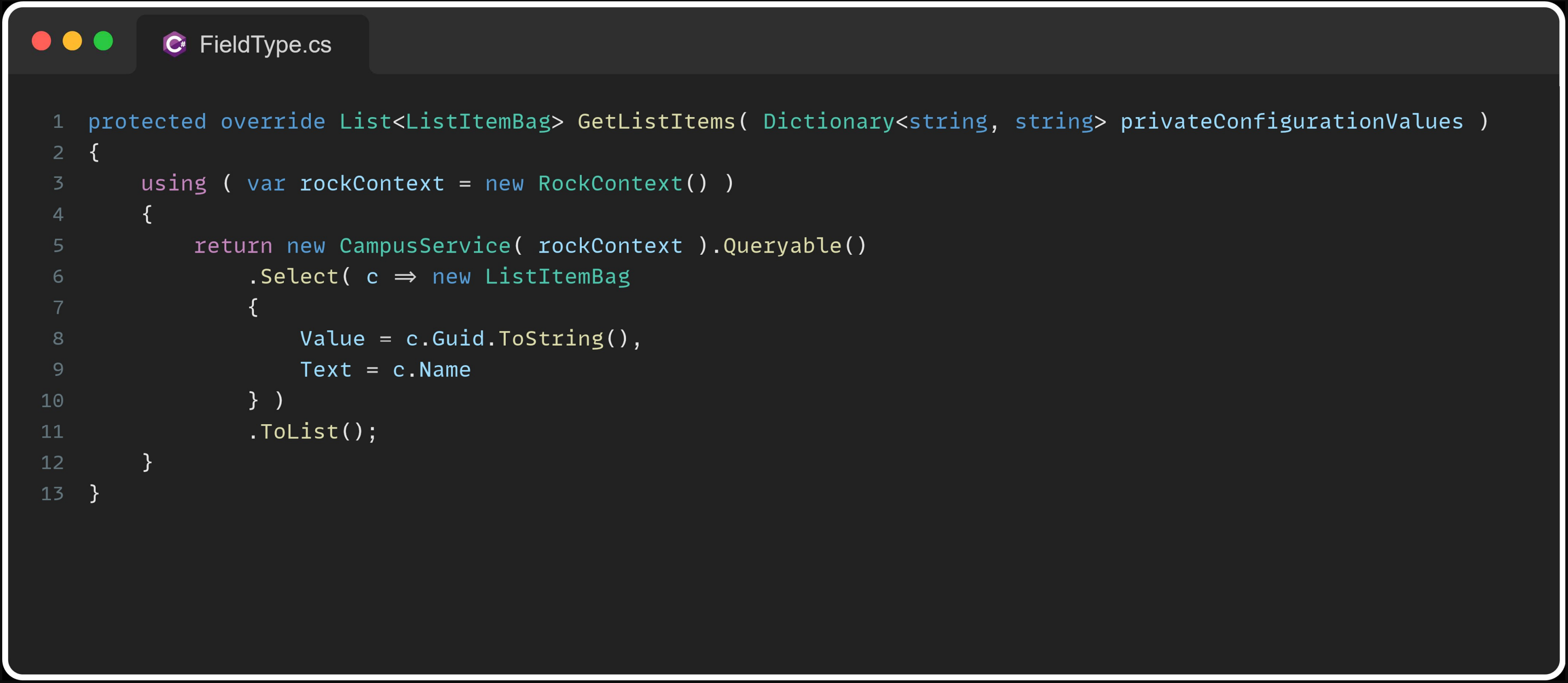
```
[FieldTypeGuid( "30EE533F-C11E-4846-AA79-1FCD96B37ACE" )]
public class TestFieldType : UniversalItemPickerFieldType
{
    protected override bool IsMultipleSelection => false;

    protected override UniversalItemValuePickerDisplayStyle GetDisplayStyle(
        Dictionary<string, string> privateConfigurationValues );

    protected override List<ListItemBag> GetListItems(
        Dictionary<string, string> privateConfigurationValues );

    protected override List<ListItemBag> GetItemBags(
        IEnumerable<string> values,
        Dictionary<string, string> privateConfigurationValues );
}
```

Item Field Type


FieldType.cs

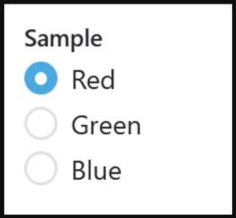
```
protected override List<ListItemBag> GetItemBags( IEnumerable<string> values,
    Dictionary<string, string> privateConfigurationValues )
{
    using ( var rockContext = new RockContext() )
    {
        return new CampusService( rockContext ).Queryable()
            .Where( c => values.Contains( c.Guid.ToString() ) )
            .Select( c => new ListItemBag
            {
                Value = c.Guid.ToString(),
                Text = c.Name
            } )
            .ToList();
    }
}
```

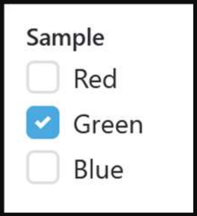
Item Field Type

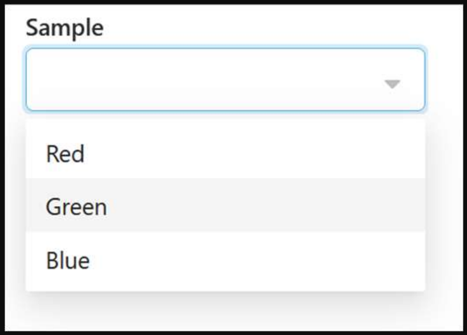

FieldType.cs

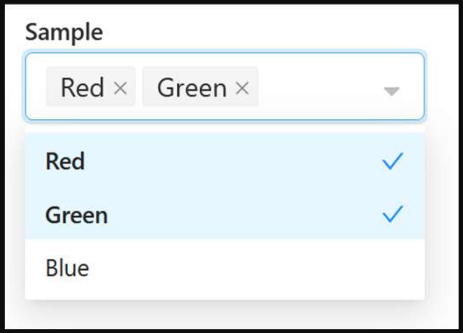
```
protected override List<ListItemBag> GetListItems( Dictionary<string, string> privateConfigurationValues )
{
    using ( var rockContext = new RockContext() )
    {
        return new CampusService( rockContext ).Queryable()
            .Select( c => new ListItemBag
            {
                Value = c.Guid.ToString(),
                Text = c.Name
            } )
            .ToList();
    }
}
```

Item Field Type

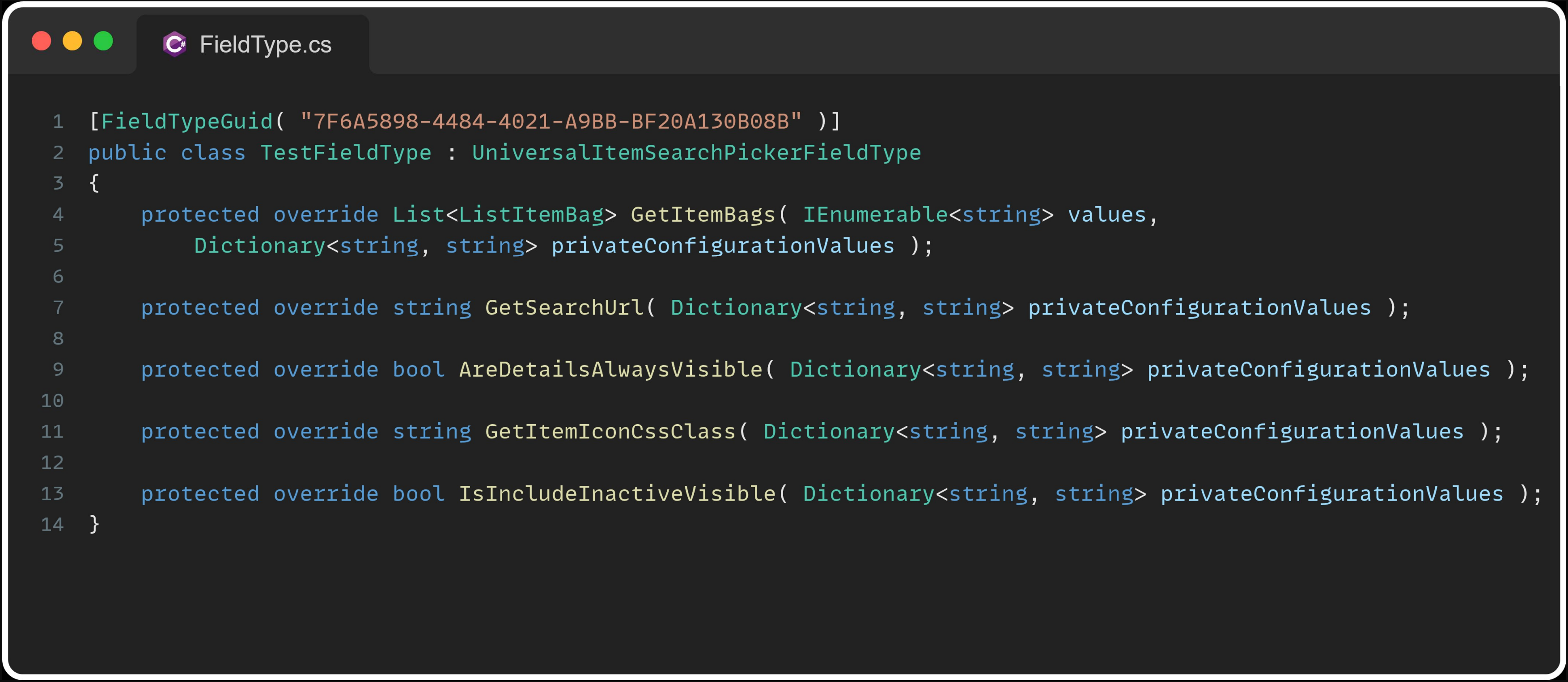






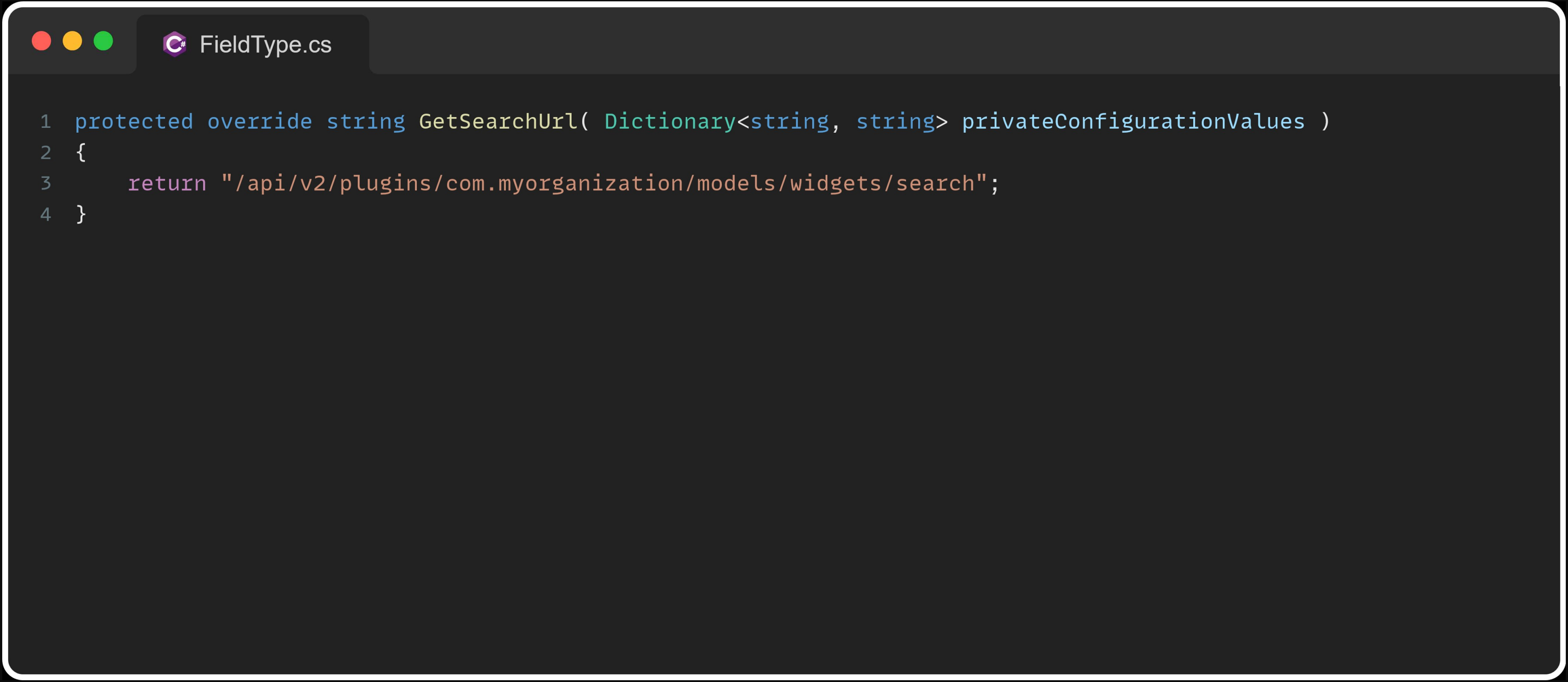


Search Field Type


FieldType.cs

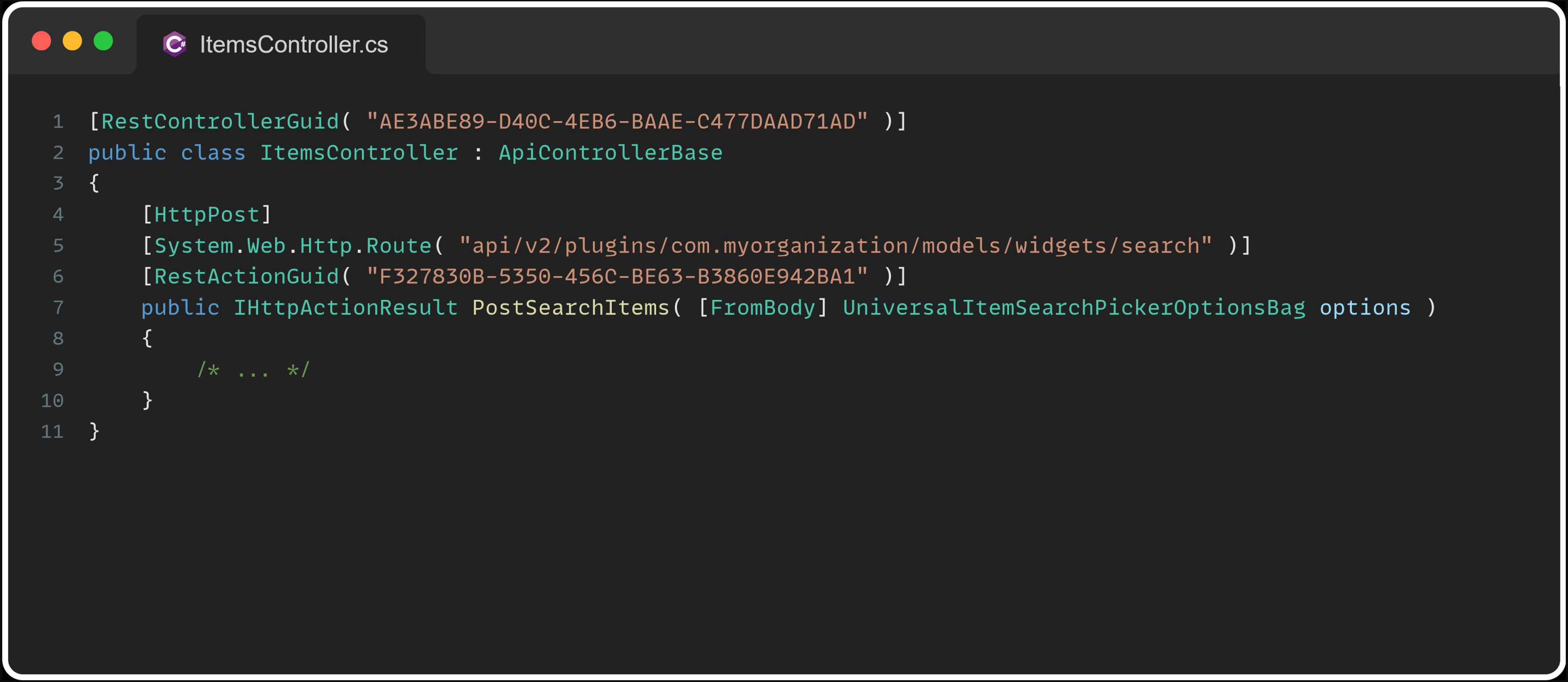
```csharp
[FieldTypeGuid( "7F6A5898-4484-4021-A9BB-BF20A130B08B" )]
public class TestFieldType : UniversalItemSearchPickerFieldType
{
    protected override List<ListItemBag> GetItemBags( IEnumerable<string> values,
        Dictionary<string, string> privateConfigurationValues );

    protected override string GetSearchUrl( Dictionary<string, string> privateConfigurationValues );

    protected override bool AreDetailsAlwaysVisible( Dictionary<string, string> privateConfigurationValues );

    protected override string GetItemIconCssClass( Dictionary<string, string> privateConfigurationValues );

    protected override bool IsIncludeInactiveVisible( Dictionary<string, string> privateConfigurationValues );
}
```

Search Field Type



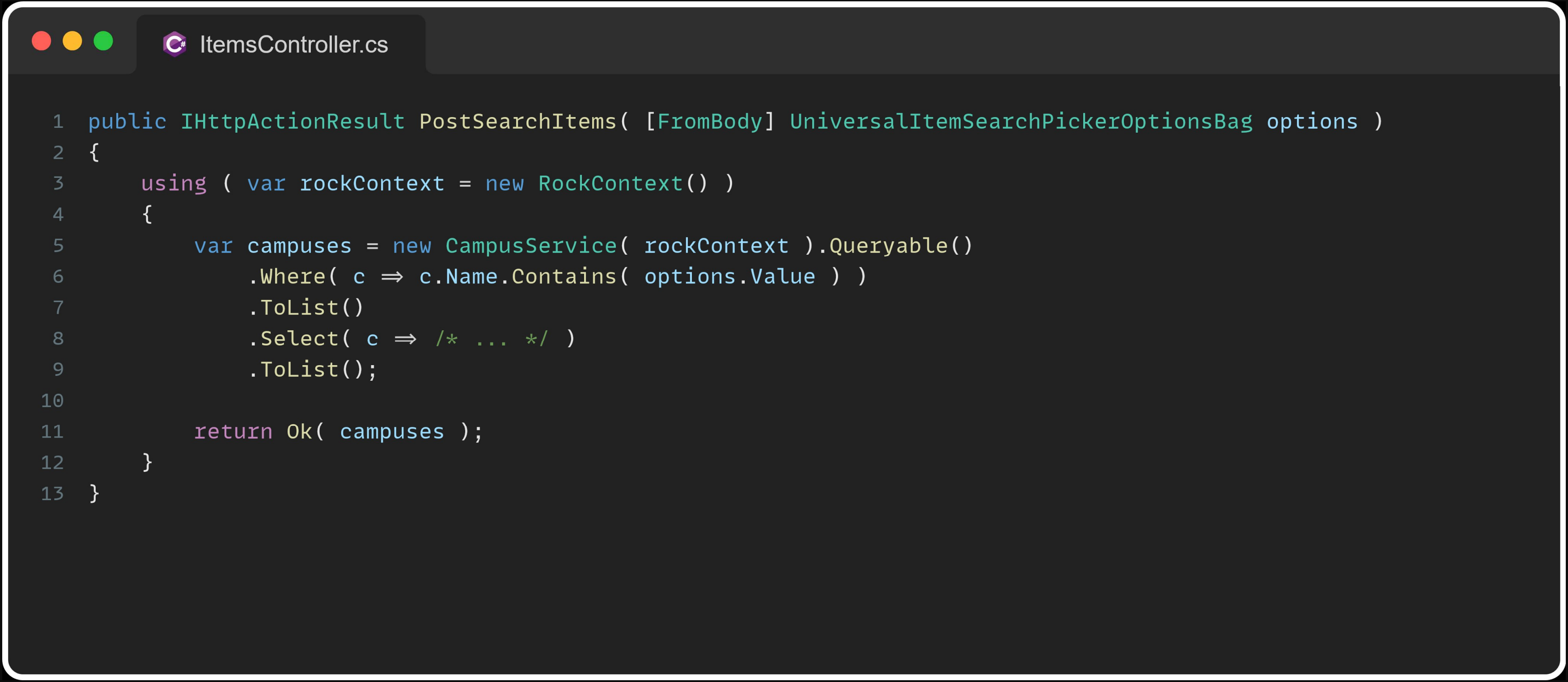
```csharp
protected override string GetSearchUrl( Dictionary<string, string> privateConfigurationValues )
{
    return "/api/v2/plugins/com.myorganization/models/widgets/search";
}
```

Search Field Type


ItemsController.cs

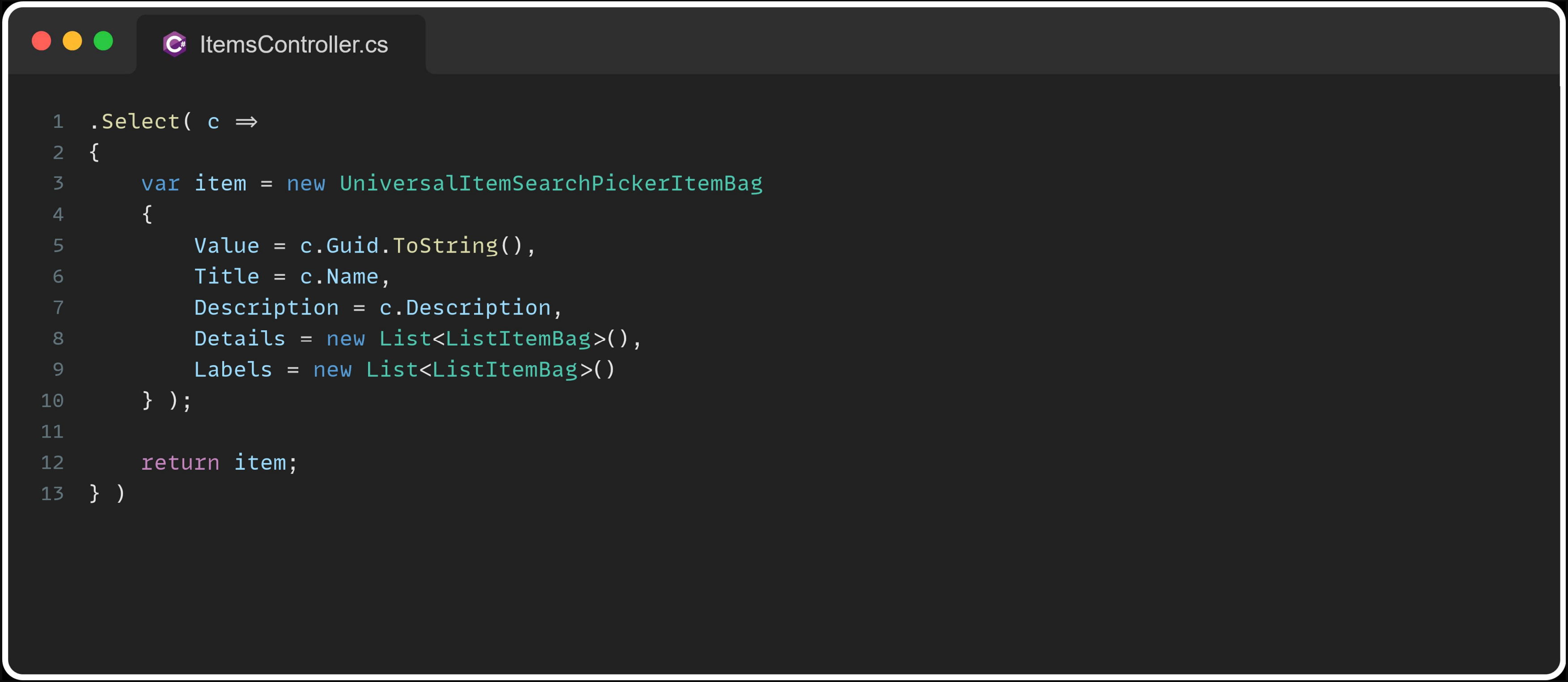
```csharp
[RestControllerGuid( "AE3ABE89-D40C-4EB6-BAAE-C477DAAD71AD" )]
public class ItemsController : ApiControllerBase
{
    [HttpPost]
    [System.Web.Http.Route( "api/v2/plugins/com.myorganization/models/widgets/search" )]
    [RestActionGuid( "F327830B-5350-456C-BE63-B3860E942BA1" )]
    public IHttpActionResult PostSearchItems( [FromBody] UniversalItemSearchPickerOptionsBag options )
    {
        /* ... */
    }
}
```

Search Field Type


ItemsController.cs

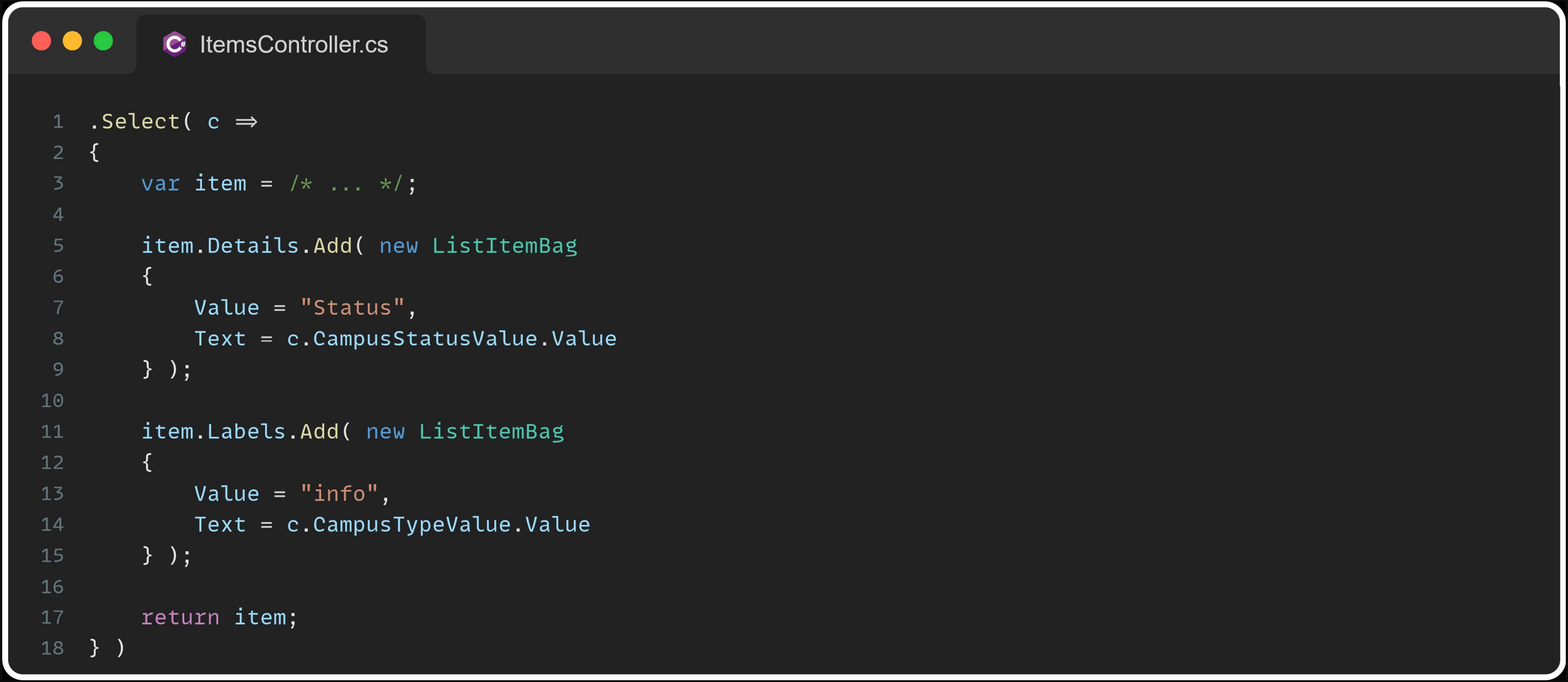
```csharp
public IHttpActionResult PostSearchItems( [FromBody] UniversalItemSearchPickerOptionsBag options )
{
    using ( var rockContext = new RockContext() )
    {
        var campuses = new CampusService( rockContext ).Queryable()
            .Where( c => c.Name.Contains( options.Value ) )
            .ToList()
            .Select( c => /* ... */ )
            .ToList();

        return Ok( campuses );
    }
}
```

Search Field Type


ItemsController.cs

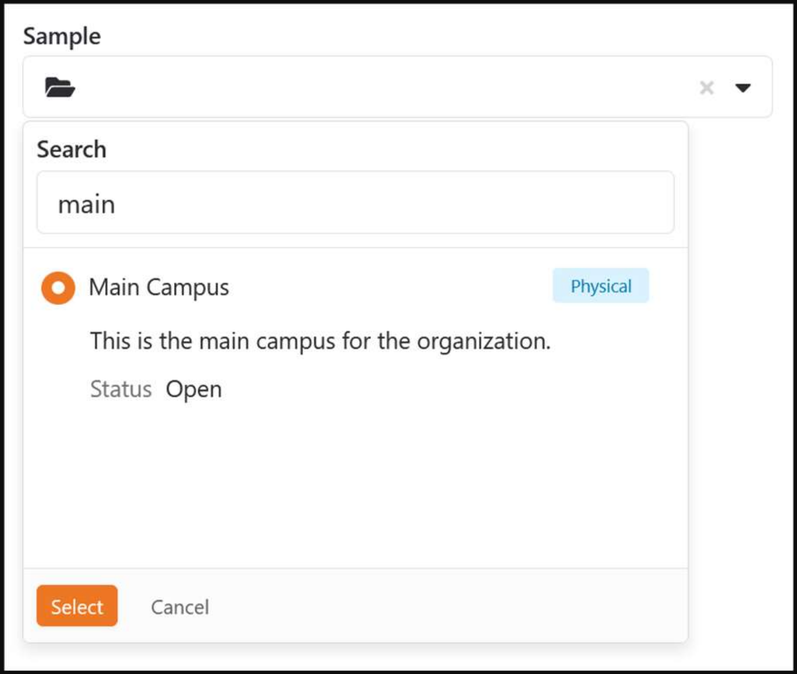
```
.Select( c =>
{
    var item = new UniversalItemSearchPickerItemBag
    {
        Value = c.Guid.ToString(),
        Title = c.Name,
        Description = c.Description,
        Details = new List<ListItemBag>(),
        Labels = new List<ListItemBag>()
    } );

    return item;
} )
```

Search Field Type

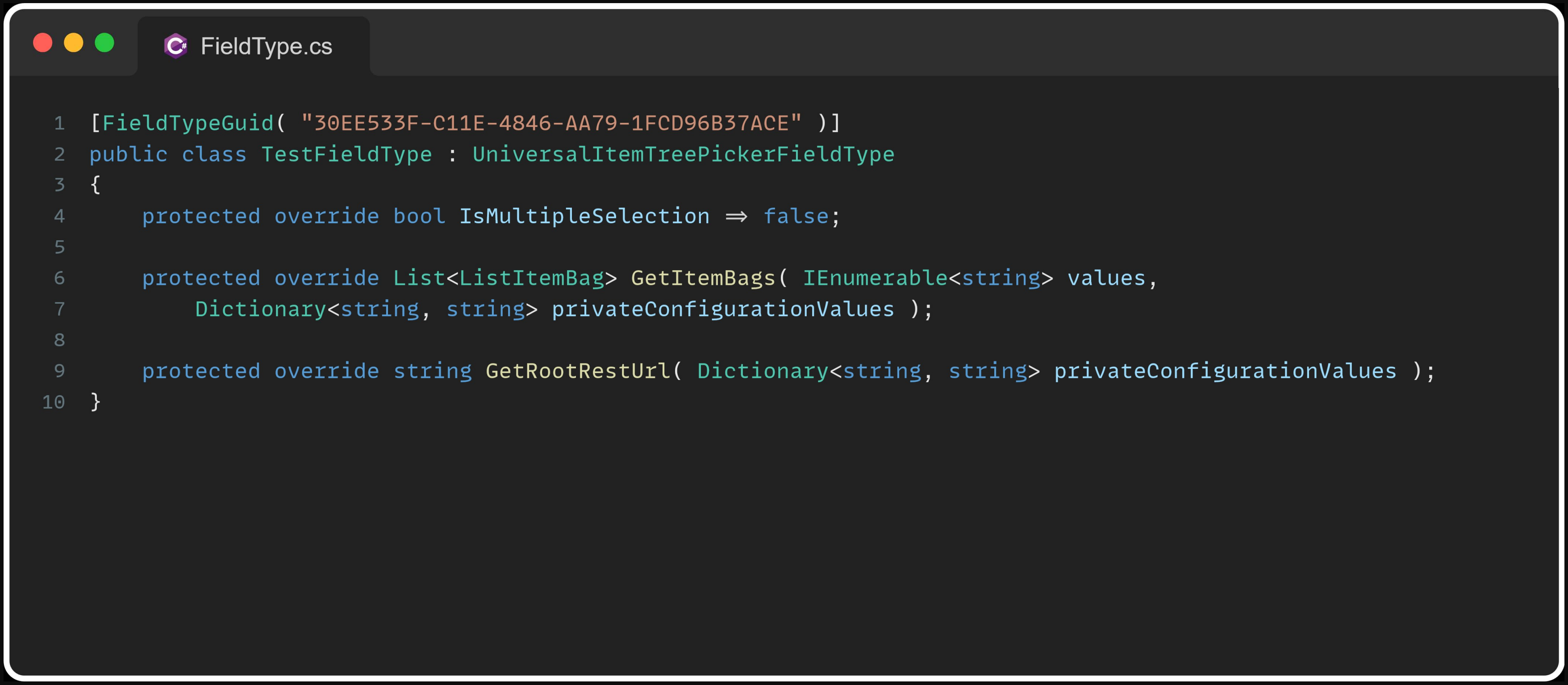


```csharp
.Select( c =>
{
    var item = /* ... */;

    item.Details.Add( new ListItemBag
    {
        Value = "Status",
        Text = c.CampusStatusValue.Value
    } );

    item.Labels.Add( new ListItemBag
    {
        Value = "info",
        Text = c.CampusTypeValue.Value
    } );

    return item;
} )
```

Search Field Type

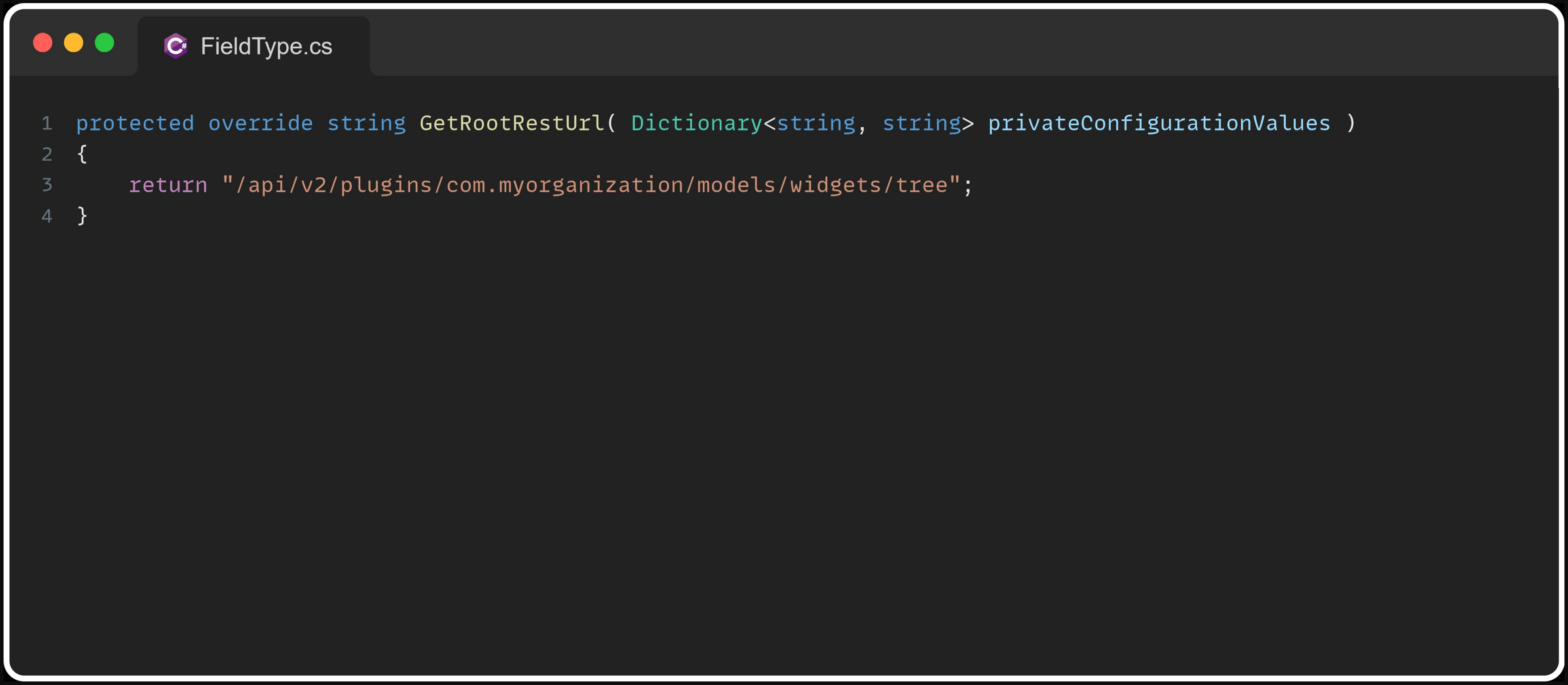


Tree Field Type


FieldType.cs

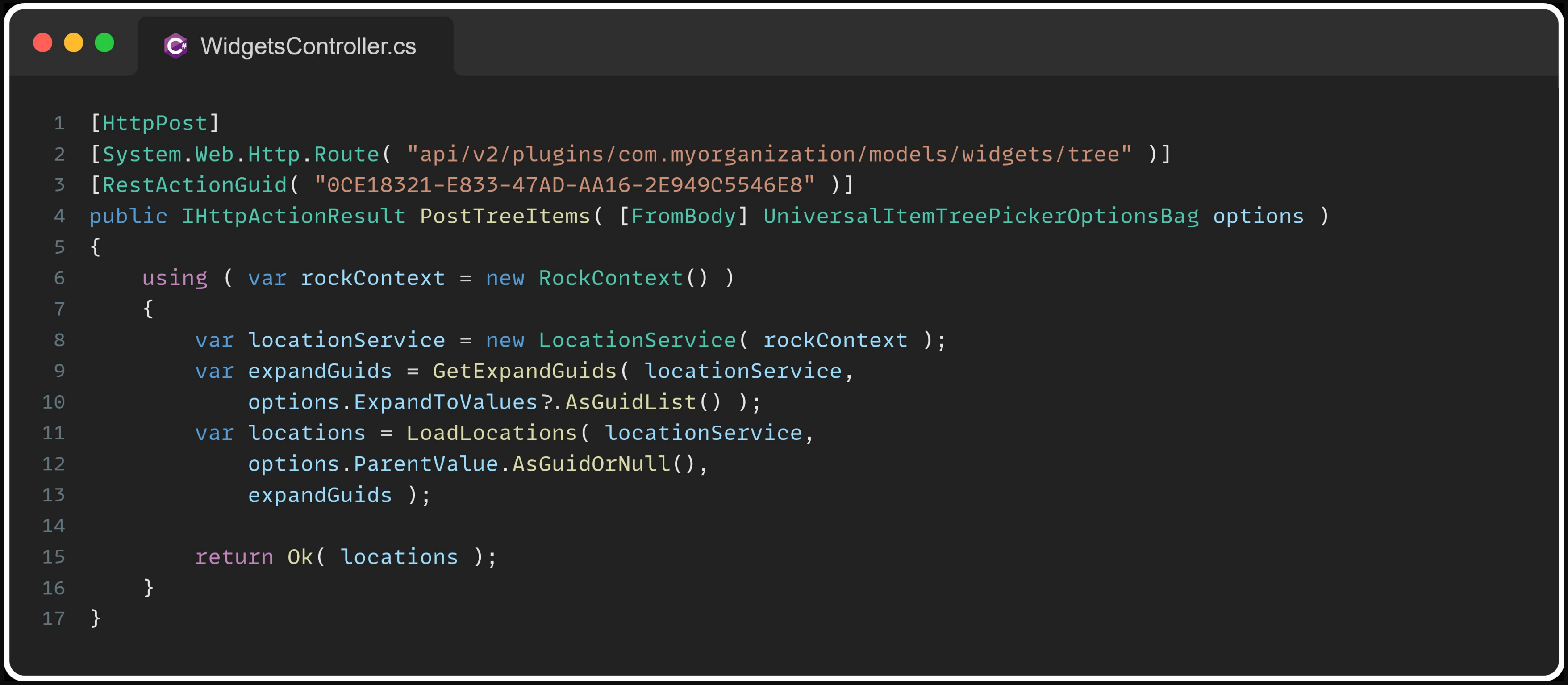
```
[FieldTypeGuid( "30EE533F-C11E-4846-AA79-1FCD96B37ACE" )]
public class TestFieldType : UniversalItemTreePickerFieldType
{
    protected override bool IsMultipleSelection => false;

    protected override List<ListItemBag> GetItemBags( IEnumerable<string> values,
        Dictionary<string, string> privateConfigurationValues );

    protected override string GetRootRestUrl( Dictionary<string, string> privateConfigurationValues );
}
```

Tree Field Type


FieldType.cs

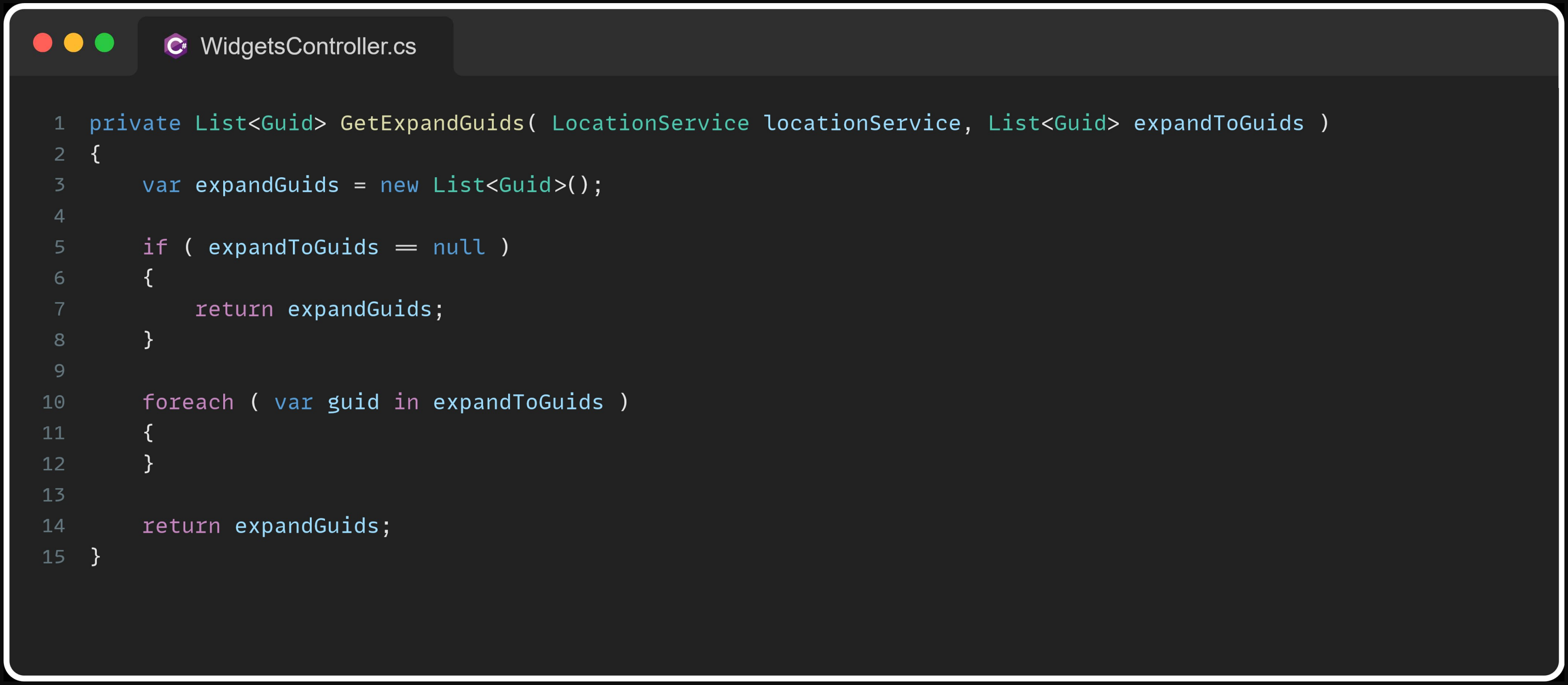
```
protected override string GetRootRestUrl( Dictionary<string, string> privateConfigurationValues )
{
    return "/api/v2/plugins/com.myorganization/models/widgets/tree";
}
```

Tree Field Type


WidgetsController.cs

```
[HttpPost]
[System.Web.Http.Route( "api/v2/plugins/com.myorganization/models/widgets/tree" )]
[RestActionGuid( "0CE18321-E833-47AD-AA16-2E949C5546E8" )]
public IHttpActionResult PostTreeItems( [FromBody] UniversalItemTreePickerOptionsBag options )
{
    using ( var rockContext = new RockContext() )
    {
        var locationService = new LocationService( rockContext );
        var expandGuids = GetExpandGuids( locationService,
            options.ExpandToValues?.AsGuidList() );
        var locations = LoadLocations( locationService,
            options.ParentValue.AsGuidOrNull(),
            expandGuids );

        return Ok( locations );
    }
}
```

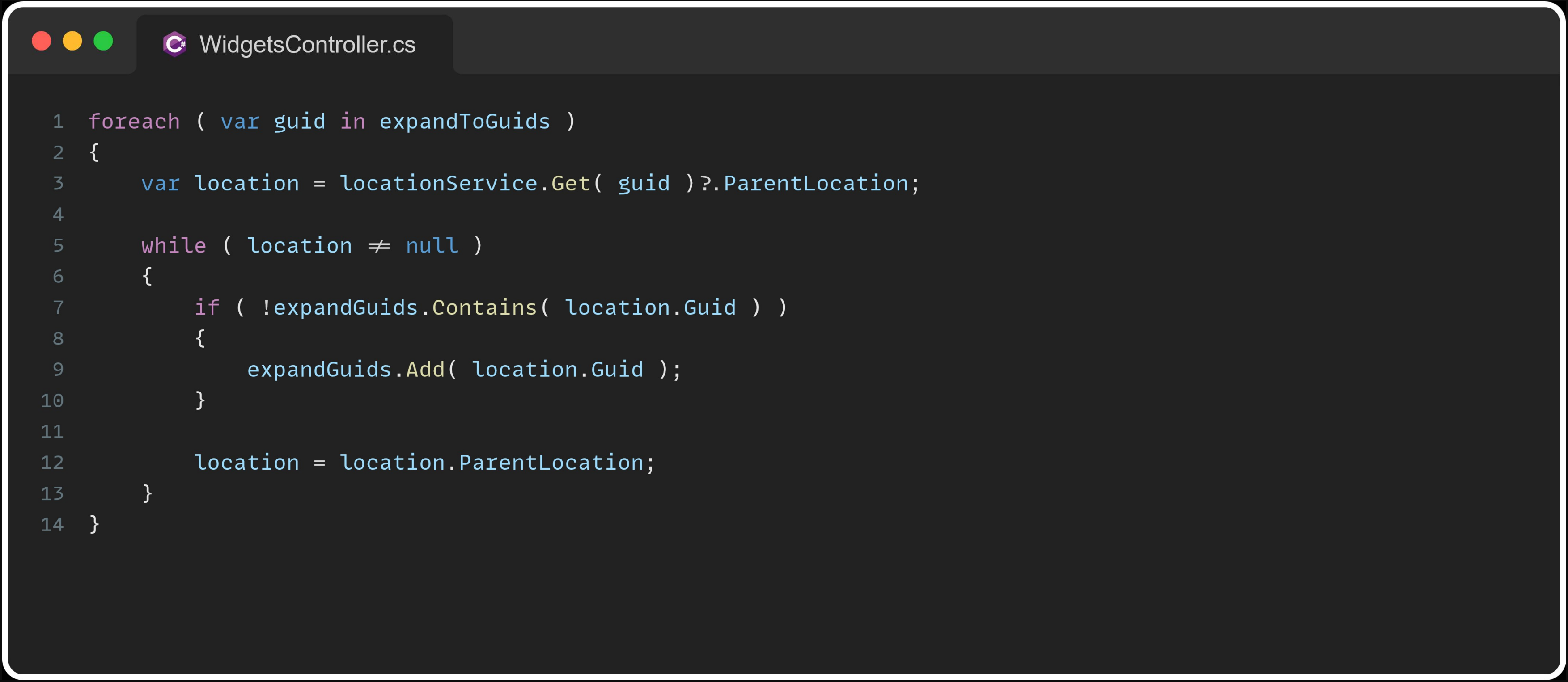
Tree Field Type



WidgetsController.cs

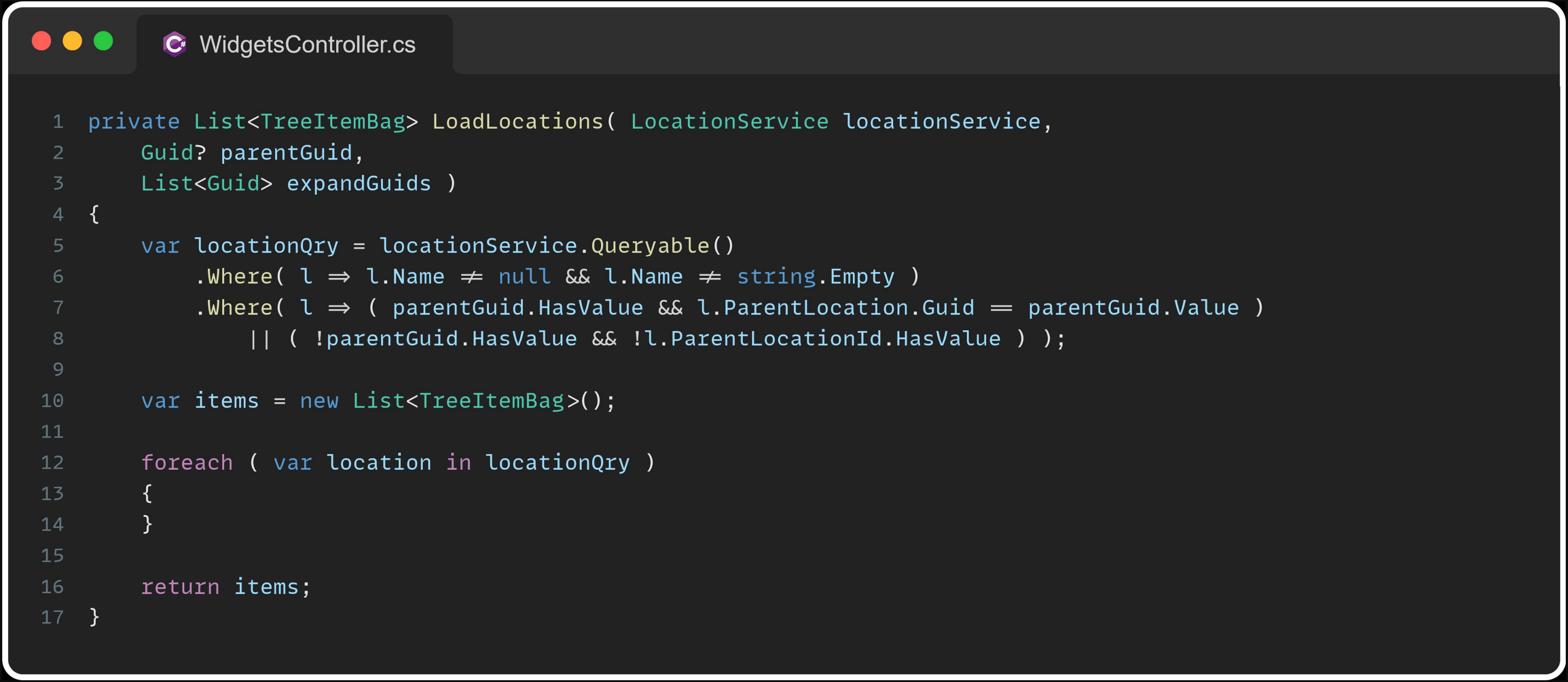
```
private List<Guid> GetExpandGuids( LocationService locationService, List<Guid> expandToGuids )
{
    var expandGuids = new List<Guid>();

    if ( expandToGuids == null )
    {
        return expandGuids;
    }

    foreach ( var guid in expandToGuids )
    {
    }

    return expandGuids;
}
```

Tree Field Type


WidgetsController.cs

```
foreach ( var guid in expandToGuids )
{
    var location = locationService.Get( guid )?.ParentLocation;

    while ( location ≠ null )
    {
        if ( !expandGuids.Contains( location.Guid ) )
        {
            expandGuids.Add( location.Guid );
        }

        location = location.ParentLocation;
    }
}
```

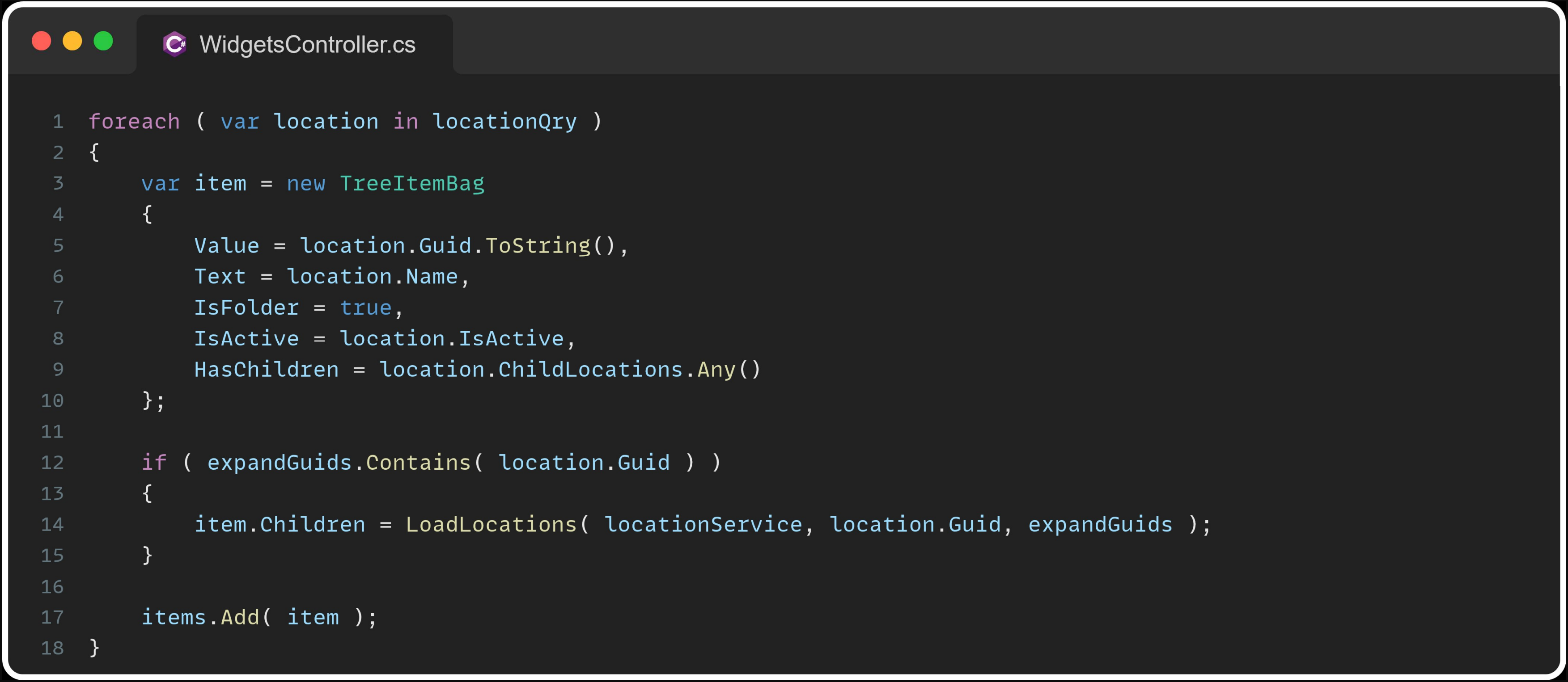
Tree Field Type



WidgetsController.cs

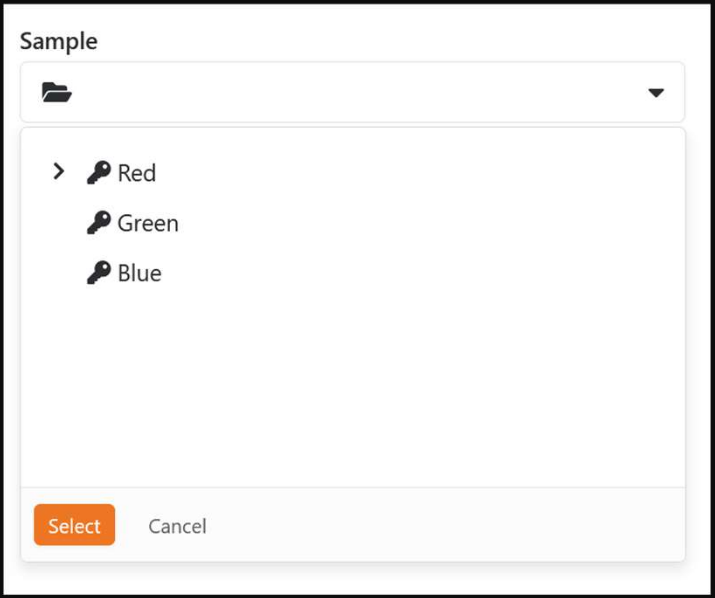
```csharp
private List<TreeItemBag> LoadLocations( LocationService locationService,
    Guid? parentGuid,
    List<Guid> expandGuids )
{
    var locationQry = locationService.Queryable()
        .Where( l => l.Name != null && l.Name != string.Empty )
        .Where( l => ( parentGuid.HasValue && l.ParentLocation.Guid == parentGuid.Value )
            || ( !parentGuid.HasValue && !l.ParentLocationId.HasValue ) );

    var items = new List<TreeItemBag>();

    foreach ( var location in locationQry )
    {
    }

    return items;
}
```

Tree Field Type

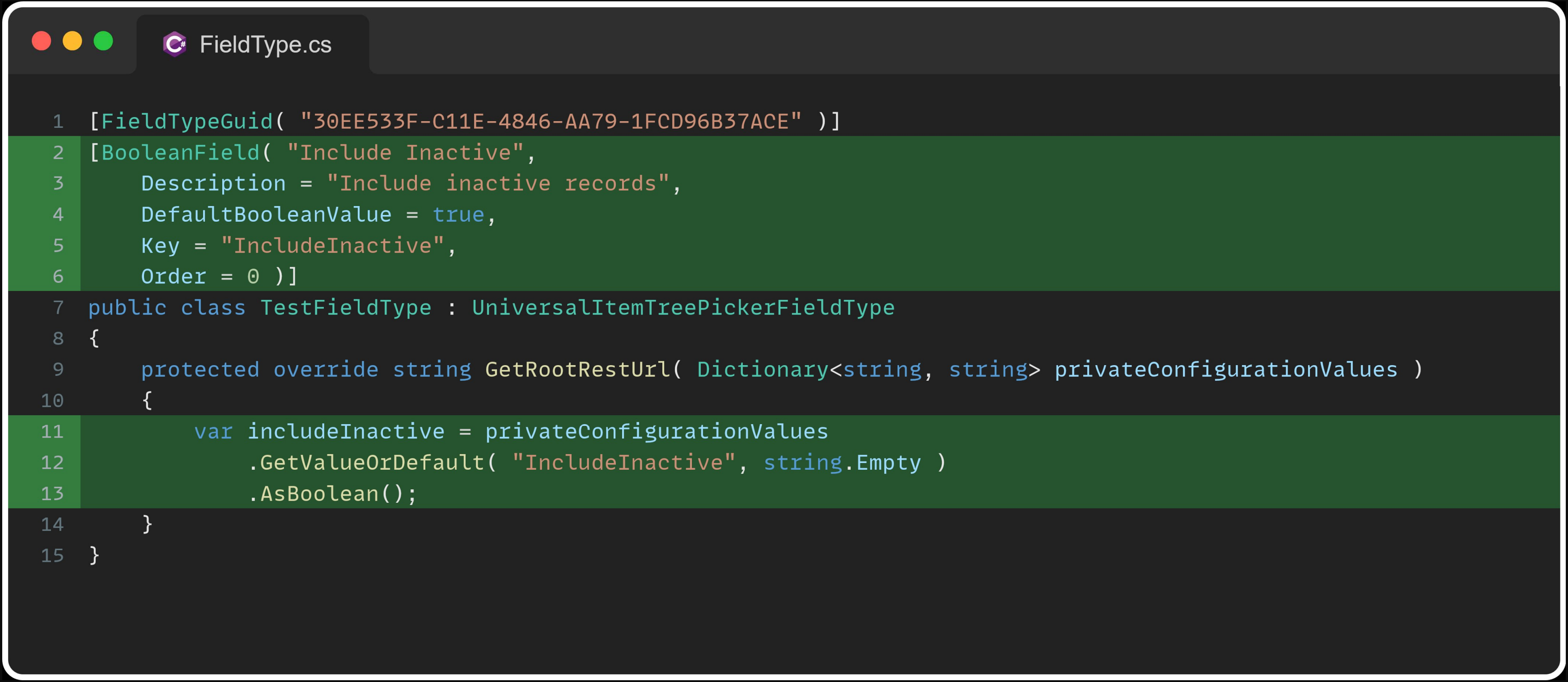

WidgetsController.cs

```
foreach ( var location in locationQry )
{
    var item = new TreeItemBag
    {
        Value = location.Guid.ToString(),
        Text = location.Name,
        IsFolder = true,
        IsActive = location.IsActive,
        HasChildren = location.ChildLocations.Any()
    };

    if ( expandGuids.Contains( location.Guid ) )
    {
        item.Children = LoadLocations( locationService, location.Guid, expandGuids );
    }

    items.Add( item );
}
```

Tree Field Type



Field Type Configuration


FieldType.cs

```
[FieldTypeGuid( "30EE533F-C11E-4846-AA79-1FCD96B37ACE" )]
[BooleanField( "Include Inactive",
    Description = "Include inactive records",
    DefaultBooleanValue = true,
    Key = "IncludeInactive",
    Order = 0 )]
public class TestFieldType : UniversalItemTreePickerFieldType
{
    protected override string GetRootRestUrl( Dictionary<string, string> privateConfigurationValues )
    {
        var includeInactive = privateConfigurationValues
            .GetValueOrDefault( "IncludeInactive", string.Empty )
            .AsBoolean();
    }
}
```

Questions?