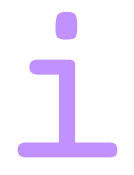
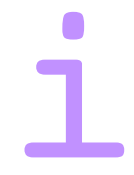
```
<h1>Counter</h1>

<p>Current count: @_currentCounter</p>

<button id="increment-button"
        class="btn btn-primary"
        @onclick="IncrementCount">Click me</button>

@code {
    private int _currentCounter { get; set; }

    private void IncrementCount()
    {
        _currentCounter++;
    }
}
```


B



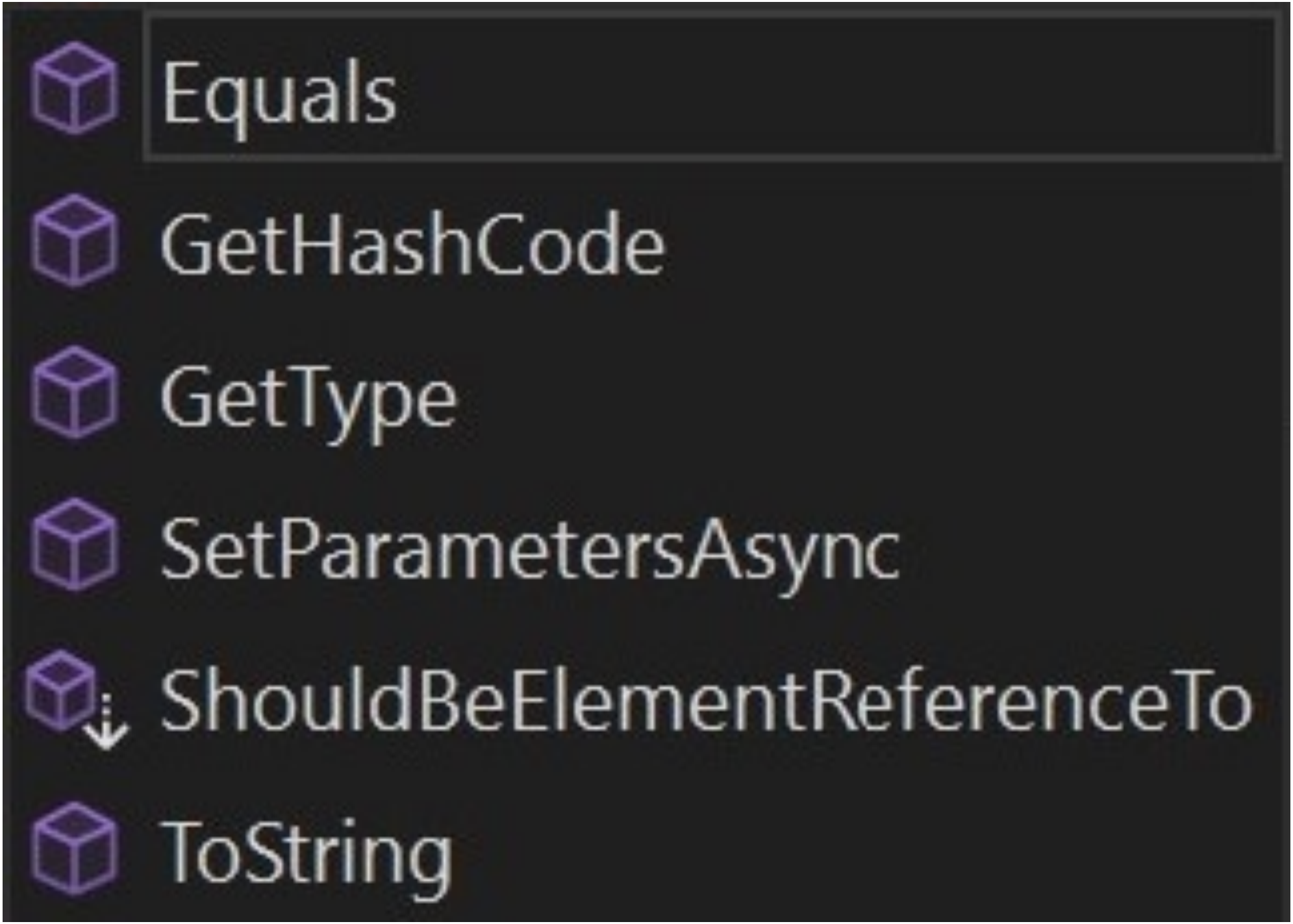


[

(


W

)


Equals
GetHashCode
GetType
SetParametersAsync
ShouldBeElementReferenceTo
ToString

```
@using Bunit
```

```
@using bUnit.Tutorial
```

```
@using Xunit
```

```
@inherits TestContext
```

@code {

[Fact]

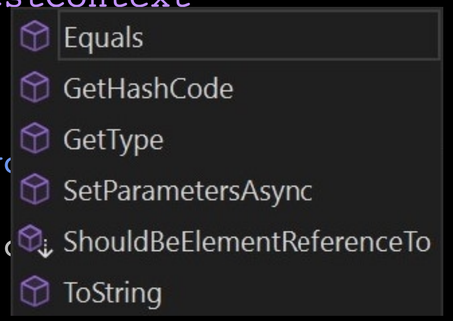
```
public void SomeTest()
```

```
var cut = new Counter();
```

cut.

Counter - Why bUnit

```razor
@using Bunit
@using bUnit.Tutorial
@using Xunit

@inherits TestContext

@code {
    [Fact]
    public vo
    {
        var c
        cut.
    }
}
```




```razor
<h1>Counter</h1>

<p>Current count: @_currentCounter</p>

<button id="increment-button"
        class="btn btn-primary"
        @onclick="IncrementCount">Click me</button>

@code {
    private int _currentCounter { get; set; }

    private void IncrementCount()
    {
        _currentCounter++;
    }
}
```

- Every Blazor component can be simply created via new()
  but component doesn't know how to render itself
- Does not help as only SetParametersAsync is exposed, the rest comes from System.Object
- Does not help with life cycle events

Counter - Why bUnit

```
@using Bunit
@using bUnit.Tutorial
@using Xunit

@inherits TestContext

@code {
    [Fact]
    public void SomeTest()
    {
        var cut = new Counter();
        cut.
    }
}
```

```
<h1>Counter</h1>

<p>Current count: @_currentCounter</p>

<button id=“increment-button"
        class="btn btn-primary”
        @onclick="IncrementCount">Click me</button>

@code {
    private int _currentCounter { get; set; }

    private void IncrementCount()
    {
        _currentCounter++;
    }
}
```