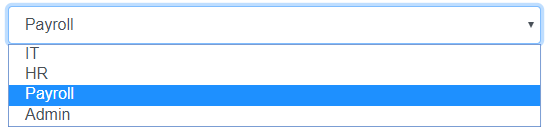
InputSelect does not support the type System.Int32

In this article we will discuss how to fix this error - InputSelect does not support the type System.Int32. This is one of the common error you will encounter when binding a select element to integer.

Blazor Select Example



<InputSelect @bind-Value="Employee.DepartmentId">

@foreach (var dept in Departments)

{

<option value="@dept.DepartmentId">@dept.DepartmentName</option>

}

</InputSelect>

Copy

In this example, the InputSelect element, is bound to DepartmentId property of the Employee object. DepartmentId is of type integer, so we get the following exception.

InputSelect does not support the type System.Int32

Only String and Enum data types are supported. Out of the box, the built-in InputSelect component only supports String and Enum data types. Integer is not supported. Since ASP.NET core is open source, we can see the source code of InputSelect on the following page.

[InputSelect Component Source Code](https://github.com/dotnet/aspnetcore/blob/58c13c312eabfb00fdb56c7e0f69813359d2fa9a/src/Components/Web/src/Forms/InputSelect.cs)

Inelegant Fix

In the component class create a string property and bind to it. The downside of this approach is we have to do the conversion twice. First from int to string while binding the value to the select element and from string to int after retrieving the selected value from the select element.

public string DepartmentId { get; set; }

Copy

Custom InputSelect

The propery way to fix this is by creating a custom InputSelect element. The easiest way to do this by inheriting the built-in InputSelect component and overriding TryParseValueFromString() method.

In the example below, we have included a case for integer data type. String and Enum data types will still be supported, because in the ELSE block we are falling back to the base class implementation of TryParseValueFromString() method.

using Microsoft.AspNetCore.Components.Forms;

namespace EmployeeManagement.Web.Pages

{

public class CustomInputSelect<TValue> : InputSelect<TValue>

{

protected override bool TryParseValueFromString(string value, out TValue result,

out string validationErrorMessage)

{

if (typeof(TValue) == typeof(int))

{

if (int.TryParse(value, out var resultInt))

{

result = (TValue)(object)resultInt;

validationErrorMessage = null;

return true;

}

else

{

result = default;

validationErrorMessage =

$"The selected value {value} is not a valid number.";

return false;

}

}

else

{

return base.TryParseValueFromString(value, out result,

out validationErrorMessage);

}

}

}

}

Copy

In all our forms we can now use this CustomInputSelect component. It supports all the following types

* Int
* String and
* Enum

If you want another data type like GUID for example, just include another IF block for GUID conversion in the CustomInputSelect component class.