

Brug af ChildAction/Partial

Hvis du vil lave et udtræk af data, som bruger en model, men som skal kunne bruges på sider, som også bruger en (anden) model ... så kan du løse det med en "ChildAction" (en slags partial).

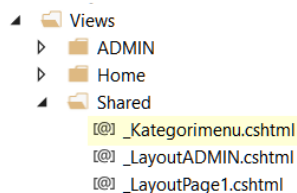
Et klassisk eksempel er fx: Du vil gerne udtrække en kategori-menu (kategori-modellen) på layoutpagen ... men layoutpagen bliver brugt til en masse sider, som bruger egne modeller, hvor kategorierne ikke er med.

1. Lav en ChildAction i en Controller – fx HomeController – og udtræk alle kategorier og send dem til den partial, som skal bruge dem

```
// Fx til brug i en partial som bliver brugt på layoutpage
[ChildActionOnly]
0 references
public ActionResult VisKategoriMenu()
{
    List<Kategori> Kategoriliste = new List<Kategori>();
    Kategoriliste = Db.Kategori.ToList();
    return PartialView("_Kategorimenu", Kategoriliste);
}
```

Husk at "return PartialView" skal return'e til den partial (her **_Kategorimenu** – den der er vist herunder), som skal bruge koden/kategorilisten.

2. Lav en "partial", som kan bruge dataene (kategorilisten) i Shared-mappen:



```
@model List<JokesWeb.Models.Kategori>

<ul>
    @foreach (var item in Model)
    {
        <li>
            <a href="~/Home/VisJokesUdfraKategori/@item.KategoriId">@item.Kategorinavn</a>
        </li>
    }
</ul>
```

3. Og brug så din ChildAction/partialview fx på layout-pagen ved at kalde "ChildAction" (ikke partial'en):

```
</head>
<body>

<div style="background-color: #EEE; padding: 10px; ">

    @Html.Action("VisKategoriMenu")

    <form action="/Home/SoegJokes" method="post">

        <input type="search" name="Soegetxt" value="" placeholder=
        <button type="submit">SØG!</button>

    </form>
```

Læs mere fx her: <https://www.codeproject.com/Articles/1079909/ASP-NET-MVC-Partial-Views-with-Partial-Models> ... Bemærk skemaet et godt stykke nede på siden, hvor de forskellige typer af "partials"/genbrugbare kodestumper bliver vist:

Partial View vs ChildAction vs EditorFor

	Centricity	Description	Uses
Partial view	View centric	Works with no model or the page model. Can work with a sub-model if it's read-only.	Nice for splitting mark-up on a complex page. e.g. Tabs. Dynamic view selection when used on layout pages.
ChildAction	View centric	Creating partial views as a controller method [ChildActionOnly]. Call using @Html.Action("action"). Can use controller logic before rendering the view. Does not use the page model.	Good for re-use of independent views on many pages.
EditorFor	Model centric	Works with simple types and complex models and maintains the element naming convention for correct post back model binding. Element naming adheres to model hierarchies.	Excellent for re-usable data model views used in forms.
DisplayFor	Model centric	By convention a read-only version of EditorFor.	
PartialFor	Model centric	Similar to EditorFor to be used with complex objects. No test for recursive objects.	With inherited or interface parts of objects. Works with collections.