Spickzettel („Cheat Sheet“) .NET Core / ASP.NET Core / Entity Framework Core

Autor: Dr. Holger Schwichtenberg Version: 1.1 / 17.3.2017

.NET Core-Kommandozeilenwerkzeuge

Neue Konsolenanwendung mit .csproj-Datei

```
dotnet new console --framework netcoreapp1.1 --language C#
```

Neues Webprojekt mit .csproj-Datei und Webvorlage

```
dotnet new mvc --framework netcoreapp1.1 --language C#
```

Liste der Vorlagen

```
dotnet new
```

Paketinstallation

```
dotnet restore
```

Übersetzen

```
dotnet build
```

Übersetzen und Starten

```
dotnet run
```

Paket erstellen

```
dotnet publish --output c:\deploy\
```

Paket ausführen

```
dotnet run c:\deploy oder dotnet run c:\deploy\anwendung.dll
```

Entity Framework Core-Kontext

```
public class FlugContext : Microsoft.EntityFrameworkCore.DbContext
 {
  public virtual DbSet<Flug> Flug { get; set; }
  protected override void OnConfiguring(DbContextOptionsBuilder
ob)  { // Provider und Connection String festlegen
  ob.UseSqlServer(@"Server=Server123;Database=FlugDB;
  Trusted_Connection=True;MultipleActiveResultSets=True;");
  }
  protected override void OnModelCreating(ModelBuilder mb)
  {
   mb.Entity<Flug>(entity =>  { // Entität konfigurieren
    entity.HasKey(e => e.FlugNr).HasName("PK_Flug");
    entity.HasIndex(e => e.Guid).IsUnique();  … }
  }
}
```

Startup-Code für ASP.NET Core-Webanwendungen

```
public class Startup
 { …
  public void Configure(IApplicationBuilder app,
  IHostingEnvironment env, ILoggerFactory loggerFactory)
 {  if (env.IsDevelopment()) {
    app.UseDeveloperExceptionPage();
    app.UseDatabaseErrorPage();
    app.UseBrowserLink();  }
   else {
    app.UseExceptionHandler("/Home/Error");
   }
 app.UseDefaultFiles();
 app.UseStaticFiles();
 app.UseSession();
 app.UseResponseCaching();
 app.UseIdentity();
 var options = new RewriteOptions().AddRewrite(
     "FlugPlan/(.*)/Paging/(.*)/(.*)", "Flug/$1/$2/$3", true);
 app.UseRewriter(options);
 app.UseMvc(routes =>{
  routes.MapRoute( // Standardroute controller/action/id?
    name: "default",
    template: "{controller=Home}/{action=Index}/{id?}");
});  } }
```

POCO-Controller

```
public class POCOController
{ // Aufruf = Standardroute: /poco/index
 public string Index()
 {   return System.DateTime.Now + ": Hallo Welt!";  }
 // Aufruf = Standardroute: /poco/hallo?name=HS
 public ActionResult Hallo(string name)
 {   return new ContentResult()   {
     ContentType = "text/html",
     Content = "<html><body><h2>Hallo " + name
     + "</h2></body></html>"   };
 }
}
```

Controller mit View

```
public class FlugController : Microsoft.AspNetCore.Mvc.Controller
 {
 // verwendet einen Entity Framework-basierten Kontext
 private FlugContext ctx = new FlugContext();
 public void Dispose() => ctx.Dispose();

 // Aufruf = Standardroute: /Flug
 public ActionResult Index()
 {
  ViewBag.GesamtAnzahl = ctx.Flug.Count();
  var flugSet = ctx.Flug.OrderBy(x=>x.Datum).ToList();
  return View(flugSet); // zeigt /views/flug/index.cshtml
 }

 // Aufruf = Standardroute: /Flug/Details/123
 public ActionResult Details(int id)
 {
 var flug = ctx.Flug.Find(id); // lädt per PK
  return View(flug); // zeigt /views/flug/details.cshtml
 }

 // Aufruf: /Flug/Rom/10/20
 [Route("/flug/{abflugort}/{von:int}/{bis:int}")]
 public ActionResult Index2(string abflugort, int von, int bis)
 {
  ViewBag.GesamtAnzahl = ctx.Flug.Count();
  var flugSet = ctx.Flug.Where(x=>x.Abflugort == abflugort)
                .Skip(von).Take(bis-von).ToList();
  return View("Index",flugSet); // zeigt /views/flug/index.cshtml
 }

 // Aufruf: http://meinServer/Flug/PlaetzeAendern/101/-1
 [Route("/flug/PlaetzeAendern/{guid}/{anzahl}")]
 public ActionResult PlaetzeAendern(string guid, short anzahl)
 {
  var flug = ctx.Flug.FirstOrDefault(x => x.Guid == guid);
  if (flug == null) return View("FlugNichtGefunden");
  flug.FreiePlaetze += anzahl;
  ctx.SaveChanges();
  return RedirectToAction("Index"); // weiter mit Index()
 }

 // Formulareinsendung mit POST
 [HttpPost]
 public ActionResult Neu([FromForm] Flug flug)
 {
  if (ModelState.IsValid) {
   ctx.Flug.Add(flug);
   ctx.SaveChanges();   }
  // weiter mit Home() in Controller Menu
  return RedirectToAction("Menu", "Home");
 }
}
```

WebAPI-Controller

```
[Route("api/[controller]")]
public class FlugApiController : Controller
{
 private FlugContext ctx = new FlugContext();
 // Aufruf /api/flugapi/123
 [HttpGet("{id}")]
 public Flug Get(int id)
 {  return ctx.Flug.Find(id);  }

 // Aufruf api/fluege/Paris/Madrid
 [HttpGet("api/fluege/{abflugOrt}/{zielort}")]
 public List<Flug> Route(string abflugOrt, string zielOrt) { … }
}
```

Spickzettel („Cheat Sheet“) .NET Core / ASP.NET Core / Entity Framework Core

Autor: Dr. Holger Schwichtenberg Version: 1.1 / 17.3.2017

Modellbasierter View

```
@using GO
@model System.Collections.Generic.List<Flug>

@Model.Count von @ViewBag.GesamtAnzahl Flügen<br>
Stand: @DateTime.Now<br>

@foreach (Flug f in Model)
{ <div> Flug @f.FlugNr fliegt am
  @f.Datum.ToString("dd.MM.yy ", new CultureInfo("de-DE")):
  @String.Format("{0:000}", f.FreiePlaetze) freie Plätze
  <a href="/flug/PlaetzeAendern/@f.Guid/-1">Buchen</a>
 </div>
}
```

Ausgaben vor allen Views mit _ViewStart.cshtml

```
@{ Layout = "_Layout"; }
```

Globale Imports mit _ViewImports.cshtml

```
@using GO
@using System.Globalization
@addTagHelper *, Microsoft.AspNetCore.Mvc.TagHelpers
```

Razor-Ausdrücke

Ausgabe ist HTML Encoded

```
@ausgabe
```

Ausgabe ist nicht Encoded

```
@Html.Raw(ausgabe)
```

Durch das Leerzeichen ist der Punkt reiner Text.

```
Dies ist die @anz. Seite!
```

Hier werden jeweils Klammern @(…) gebraucht!

```
Dies ist die @(anz).Seite!
Ergebnis: @(anz * 10 + 2)
@(new Autor().GetInfo<string>())
```

Escape für @

```
Die Variable @@anz enthält @anz.
```

Variantendeklaration

```
@{int gesamt = ViewBag.GesamtAnzahl; }
```

Razor-Ausdrücke in HTML-Attributen

Wenn cssklasse null ist, verschwindet das class-Attribut:

```
@{ string cssklasse = "intro"; }
<div class="@cssklasse">Text</div>
```

Aber nicht, wenn es weitere Werte gibt:

```
<div class="@cssklasse kursiv">Text</div>
```

Aus true wird hier checked="checked":

```
@{ var aktiv = true; }
<input type="checkbox" checked="@aktiv" />
```

Razor-Blöcke

```
@* Dies ist ein
 Kommentarblock *@

@{  // Codeblock
 int anz = 5;  string ausgabe = "<i>kursiv</i>";
 var orte = new List<string>() { "Essen","Dortmund","Bochum" };
}

@{  // Codeblock mit Markup-Inhalt
  int schrittweite = 2;
  for (int i = 0; i < anz; i += schrittweite)
  {  <div>@start.AddDays(i)</div>  }
}

@{ // Reiner Text im Codeblock
 for (int i = 0; i < anz; i += schrittweite)
 {  <text> Seite @i
  noch mehr reiner Text
  </text> }
}

@{  // Reiner Text im Codeblock (Alternative Schreibweise)
 for (int i = 0; i < anz; i += schrittweite)
 { @: Seite @i
  @: noch mehr reiner Text
 } // Klammer muss in diesem Fall in getrennte Zeile!
}
```

Razor-Schleifen

for-Schleife

```
@for (int i = 0; i < anz; i++)
{ <div>@start.AddDays(i)</div> }
```

while-Schleife

```
@while (anz*10 < 100)
{ anz++;  <div>@(anz*10)% erledigt</div> }
```

foreach-Schleife

```
@foreach (var ort in orte)
{ <a href="/flug/@ort/0/10">@ort</a> }
```

Razor-Bedingungen

```
@if (anz > 100)
{  <text>reiner Text</text> }
else
{  <div>Tag und Ausdruck @anz</div> }
```

Razor-Funktionen

```
@functions
{
 string GetProzent(int anz)
 {  return anz.ToString() + "%";  }
}
Erledigt sind @GetProzent(10);
```

HTML-Helper für Formulare

```
@{ var orteItems = orte.Select(x => new SelectListItem { Text = x,
Value = x }) .ToList();  }
@using (Html.BeginForm("Neu", "Flug", FormMethod.Post)){
 @Html.ValidationSummary("Fehleingaben:")
  <div> FlugNr @Html.TextBoxFor(x => x.FlugNr, new { @class =
"form-control", disabled="disabled" })
  @Html.ValidationMessageFor(x => x.FlugNr, "", new { @class =
"text-danger" })
 }
  Abflugort: @Html.DropDownListFor(x => x.Abflugort, orteItems,
  "offen", new { @class = "form-control" }) <br />
  Nichtraucherflug: @Html.CheckBoxFor(x => x.NichtRaucherFlug,
  new { @class = "form-control" }) <br />
  <button type="submit">Speichern</button></div> }
```

Weitere: @Html.ActionLink(), @Html.Hidden(), @Html.ListBox(), @Html.Password(), @Html.RadioButton(), @Html.TextArea()
Modellgetrieben: @Html.EditorFor(), @Html.ValueFor(), @Html.DisplayFor(), @Html.LabeFor()

Tag-Helper (nur ASP.NET Core)

```
<form asp-action="Neu" asp-controller="Flug" method="POST">
<div asp-validation-summary="All" class="text-danger"></div>
FlugNr <input asp-for="FlugNr" class="form-control"/>
<span asp-validation-for="FlugNr"></span>
Abflugort:
<select asp-for="Abflugort" asp-items="orteItems" ></select>
Nichtraucherflug:
<input type="checkbox" asp-for="NichtRaucherFlug"  />
<button type="submit" >Speichern</button>
</form>
```

Weitere:

```
<environment names="dev,test">,
<a asp-controller="Home" asp-action="Index">,
<img src="~/HS.jpg" asp-append-version="true" />,
<cache expires-after="@TimeSpan.FromMinutes(10)">
```

Über den Autor

Dr. Holger Schwichtenberg gehört zu den bekanntesten Experten für die Programmierung mit Microsoft-Produkten in Deutschland. Er hat zahlreiche Bücher zu .NET und Webtechniken veröffentlicht und spricht regelmäßig auf Fachkonferenzen wie der BASTA. Sie können ihn und sein Team für Schulungen, Beratungen und Projekte buchen.
E-Mail: **anfragen@IT-Visions.de** Website: **www.IT-Visions.de**
Weblog: **www.dotnet-doktor.de** Twitter: **@dotnetdoktor**