



HTMX E WEBSTENCILS

Un approccio lowcode alla programmazione web



LUCA MINUTI



luccominuti.it



luca.minuti@gmail.com



dev.to/lminuti



github.com/lminuti



linkedin.com/in/luccominuti/



DelphiDay
italian conference

20 Novembre 2024
Padova



wintech
italia

OPEN-SOURCE PROJECTS

github.com/Iminuti

WiRL

github.com/delphi-blocks/WiRL

Delphi SAML

github.com/EtheaDev/Delphi-SAML

OpenSSL

github.com/Iminuti/Delphi-OpenSSL



20 Novembre 2024
Padova





AGENDA

1. Introduzione HTMX
2. Demo
3. WebStencils
4. Integrazione con WiRL



HTMX

1



PANORAMICA

- Ci sono letteralmente centinaia di framework JavaScript
 - A cosa servono?
 - Sono veramente necessari?



LIMITI DI HTML

- Il problema è che in HTML ci sono pochissime possibilità di interagire con l'utente:
 - I link: `<a>`
 - L'invio delle form: `<form>`
- Quindi l'unica possibilità di interazione passa dal caricamento di un'intera nuova pagina
- Per tutto il resto abbiamo bisogno di JavaScript



SOLUZIONE HTMX

- HTMX tramite attributi custom rende possibile:
 - Comunicazione col server
 - Manipolazione del DOM

demo time

```
<script src="https://unpkg.com/htmx.org@2.0.3"></script>  
  <!-- have a button POST a click via AJAX -->  
  <button hx-post="/clicked" hx-swap="outerHTML">  
    Click Me  
  </button>
```

Quando un utente clicca su questo pulsante, viene inviata una richiesta AJAX a /clicked, e l'intero pulsante viene sostituito con la risposta HTML



HTMX ESTENDE HTML

- Qualsiasi elemento può inviare una richiesta HTTP
- Qualsiasi evento può attivare richieste
- Qualsiasi verbo HTTP, non solo GET e POST
- Qualsiasi elemento può essere il target per l'aggiornamento della richiesta

Quando si usa HTMX il server risponde con HTML non con JSON



AJAX

- **hx-get**: Invia una richiesta GET all'URL specificato
- **hx-post**: Invia una richiesta POST all'URL specificato
- **hx-put**: Invia una richiesta PUT all'URL specificato
- **hx-patch**: Invia una richiesta PATCH all'URL specificato
- **hx-delete**: Invia una richiesta DELETE all'URL specificato



TRIGGER

- Di default HTMX usa come trigger l'evento "naturale" del elemento HTML
 - per **input**, **textarea** e **select** il trigger è l'evento "change"
 - per le **form** il "submit"
 - per tutto il resto l'evento click
- Con l'attributo **hx-trigger** è possibile modificare l'evento da usare

demo time





QUANDO USARLO

- Progetti non troppo complessi
- Buona conoscenza di HTML e CSS
- È possibile integrarlo con del JavaScript per scenari più complessi
 - Consigliato Vanilla JavaScript (ma anche framework leggeri come Alpine.js)



Server side

2



SERVER SIDE

- Il server deve restituire HTML
- È una buona idea?
- data+template=html
 - WebStencils (o librerie simili)



WEBSTENCILS

- Novità di Delphi 12.2
- Si basa su due elementi:
 - il simbolo @
 - parentesi graffe per i blocchi { }



IL SIMBOLO @

- Direttive custom
 - @person.name
 - Accede alla proprietà **name** dell'oggetto **person**.
- Direttive predefinite
 - @page, @query, @* commenti, @if @else, @forEach, @renderBody, @layoutPage, @Import, ...



I BLOCCHI

- I blocchi racchiudono una serie di direttive
- Si usano per nel caso di costrutti condizionali:
 - @if, @else
- O loop:
 - @forEach

@property

```
<p> Hello @person.name </p>
```

@if @else

```
@if not cart.isEmpty {  
    <p>You have @cart.itemCount items in your cart.</p>  
}  
@else {  
    <p>Your cart is empty.</p>  
}
```

@forEach

```
<ul>  
  @ForEach (var color in colors) {  
    <li>@color.text</li>  
  }  
</ul>
```



TWebStencilsProcessor

- Proprietà e metodi principali:
 - **InputFilename**: il template (file)
 - **InputLines**: il template (TStrings)
 - **AddVar**: aggiunge un oggetto al template
 - **Content**: applica il template

demo time





WiRL

3



WIRL

- **MessageBodyWriter:**
 - il sistema che WiRL usa per trasformare gli oggetti nel formato richiesto dal client (di solito JSON ma non solo)
 - WebStencils è un modo per trasformare oggetti in HTML



L'IDEA

- Normalmente il server restituisce il dato grezzo (JSON) e il client si occupa della sua visualizzazione
- L'idea è quella di estendere HTMX per chiedere la risorsa (es.: persona, fattura, ecc.) ma con un template applicato dal server (versione readonly, editabile, concisa, estesa)

HTMX

```
<button hx-get="rest/default/person"  
  wirl-stencil="person_it.html"  
  hx-trigger="click"  
  hx-target="#output2"  
  hx-swap="outerHTML">  
  Click Me!  
</button>
```

HTMX

```
document.body.addEventListener('htmx:configRequest', function(evt) {  
  evt.detail.headers['accept'] = 'text/html'; // force text/html  
  if (evt.target.getAttribute('wirl-stencil')) {  
    evt.detail.headers['x-stencil'] =  
      evt.target.getAttribute('wirl-stencil'); // add x-stencil header  
  }
```

@MessageBodyWriter

begin

```
LStencilName := FRequest.Headers['x-stencil'];
```

```
LStencil := TWebStencilsProcessor.Create(nil);
```

try

```
LStencil.InputFileName := TPath.Combine(ExtractFileDir(ParamStr(0)), 'www', LStencilName);
```

```
LStencil.AddVar('value', AValue.AsObject, False);
```

```
LContent := LStencil.Content;
```

```
LBuffer := TEncoding.UTF8.GetBytes(LContent);
```

```
AContentStream.WriteBuffer(LBuffer[0], Length(LBuffer));
```

finally

```
LStencil.Free;
```

end;

end;

demo time





PER SAPERNE DI PIÙ

- [Articolo di Marco Cantù](#)
- [Mio articolo su WebStencils con WiRL](#)
- [Guida su WebStencils \(PDF ~70 pagine\)](#)



THANK YOU