André Custódio

Criando Scrapers com Playwright.NET

Do Scraper Básico ao Agente de IA:

Construindo Coletores de Dados Inteligentes.

Patrocínio:

GOLD

UniSales – Centro Universitário Salesiano

ABF TECH

GGBR SOLUTIONS

LAB Consultoria

Microsoft

AMD

SILVER

globalsys

15 anos

AEVO

INSTANT TECNOLOGIA

NUV BRASIL

Apoiadores:

7ZERO2 VENDAS & INOVAÇÃO

Já credenciei

BRONZE

GGTIC-ES

Four Hands – CONEXÕES ASSERTIVAS

VIX25

www.dataes.com.br

Quem sou eu?

André Custódio

Senior Software Engineer

- Especialista em .NET, focado em sistemas de back-end escaláveis.
- Foco em soluções resilientes e integrações inteligentes.
- Entusiasta de web scraping desde os dias do HtmlAgilityPack.
- Trabalho com criação de bots/scrapers para o mercado de ingressos de eventos ao vivo.

O que é Scraping?

Scraping é o processo de extrair dados de um website de forma eficiente.

Foco no básico

- Navigate
- Analyse
- Extract
- Store

Playwright.NET

Playwright.NET é uma biblioteca para automação de navegadores em .NET, que oferece recursos para interagir com páginas, incluindo as que carregam conteúdo via JavaScript. Com Playwright.NET, é possível realizar tarefas como clicar em botões, observar comportamentos e extrair dados de forma programática, garantindo maior flexibilidade e controle sobre o scraping, especialmente em sites interativos.

Por que Playwright.NET?

O que o torna ideal para extração de dados?

- Resiliência (Auto-Waits): O Playwright espera *automaticamente* que os elementos estejam prontos. Adeus, `Task.Delay()`!
- Seletores Modernos: Focados no usuário (role, text) em vez de seletores CSS frágeis.
- Alto Desempenho: Comunicação rápida e execução paralela nativa.
- Contextos Isolados: Cria "perfis de navegador" limpos em milissegundos.

Exemplo 1: Setup

```
// Inicializa o Playwright Wrapper
using var playwright = await Playwright.CreateAsync();

// Inicializa o navegador Chromium
await using var browser = await playwright.Chromium.LaunchAsync(
    new BrowserTypeLaunchOptions { Headless = false }
);

// Cria uma nova página
var page = await browser.NewPageAsync();

// Navega para a URL da página estática
await page.GotoAsync("http://localhost:8080", new PageGotoOptions
{
    WaitUntil = WaitUntilState.NetworkIdle
});

// (...)

await page.DisposeAsync();
await browser.DisposeAsync();
await playwright.DisposeAsync();
```

Exemplo 2: Extração Simples Usando Selector

```
// Seletores CSS
var header = await page.QuerySelectorAsync("#page-header");
var books = await page.QuerySelectorAllAsync(".book-row");

// Seletor XPath
var xpathBooks = page.Locator("xpath=//div[@class='book-row']");

// Estratégia de Espera
await page.WaitForSelectorAsync(".book-card", new()
{
    State = WaitForSelectorState.Visible
});

// (...) continuação do código

await page.DisposeAsync();
await browser.DisposeAsync();
await playwright.DisposeAsync();
```

Exemplo 3: Interceptação de Respostas

```
// Captura de dados da API
await page.RouteAsync("**/api/books**", async route =>
  {
      var response = await route.FetchAsync();
      var body = await response.TextAsync();

      if (response.Headers["content-type"]?.Contains("json") == true)
      {
          var data = JsonSerializer.Deserialize<BooksResponse>(body);
          books.AddRange(data.Books);
      }

      await route.FulfillAsync(new() { Response = response });
  });
```

Exemplo 4: Tratamento de Erros

```
try
{
    //(...) Código para interagir com o elemento
}
catch (PlaywrightException ex)
{
    // Capturar screenshot em caso de erro
    var screenshotPath = $"error-playwright-{DateTime.Now:yyyyMMddHHmmss}.png";
    await page.ScreenshotAsync(new PageScreenshotOptions
    {
        Path = screenshotPath,
        FullPage = true
    });

    // Logar o erro no Seq
    Log.Error(ex, "Erro do Playwright ao interagir com {Selector}. Screenshot: {Screenshot}",
        selector, screenshotPath);

    // Re-lançar a exceção para ser tratada no código principal
    throw;
}
```

Exemplo 5:
Juntando todas as coisas até agora

Arquitetura: Page Object Model (POM)

POM é um padrão que encapsula seletores e lógica de interação em uma classe.

Qual a vantagem?

- Manutenção: Se o site muda o seletor do livro, você corrige em um lugar.
- Clareza: Seu script principal lê como uma lógica de negócio, não como comandos de automação.
- Reutilização: Facilita a extração da mesma página com parâmetros diferentes.

Exemplo 6: Page Object Models

```
0 references
class BookDetailPage(IPage page)
{
    2 references
    private readonly IPage _page = page;

    0 references
    public async Task<string> GetTitleAsync()
    {
        var title = await _page.Locator(".book-detail-title").TextContentAsync();
        return title?.Trim() ?? "";
    }

    0 references
    public async Task<List<string>> GetAuthorsAsync()
    {
        var authorElements = await _page.Locator(".book-detail-authors .author-name").AllTextContentsAsync();
        return authorElements.Select(a => a.Trim()).ToList();
    }
}
```

Exemplo 7: Agentic Scraping

```
const string BookInfoSchema = """
    public class BookInfo
    {
        public string Key { get; set; } = "";
        public string Title { get; set; } = "";
        public List<string> Authors { get; set; } = new();
        public string Isbn { get; set; } = "";
    }
    """;

using var playwright = await Playwright.CreateAsync();
using var browser = await playwright.Chromium.LaunchAsync(new BrowserTypeLaunchOptions { Headless = false });

var page = await browser.NewPageAsync();
await page.GotoAsync("http://localhost:8000", new PageGotoOptions { WaitUntil = WaitUntilState.NetworkIdle });

var pageHtml = await page.ContentAsync();
var geminiAgent = new GeminiAgentService(apiKey);

var extractedBooks = await geminiAgent.ExtractBooksFromHtmlAsync(pageHtml, BookInfoSchema);

await page.DisposeAsync();
await browser.DisposeAsync();
await playwright.DisposeAsync();
```

O que é Observabilidade?

Os 3 pilares para entender seu scraper

- **LOGS**: O que aconteceu? Quando aconteceu? Como aconteceu?
- **MÉTRICAS** : O quanto aconteceu?
- **TRACES**: Como a requisição flui? Como esperar esse erro?

Exemplo 8: Observabilidade

```
// Introduzindo métricas de performance

// Overview Dashboard Metrics
var scrapingErrorsCounter = Metrics.CreateCounter("scraping_errors_total", "Total de erros durante scraping", new[] { "error_type" });

// Response Times Dashboard Metrics
var apiResponseTimeHistogram = Metrics.CreateHistogram("api_response_time_seconds", "Tempo de resposta da API em segundos",
    new[] { "endpoint" },
    new HistogramConfiguration { Buckets = Histogram.ExponentialBuckets(0.01, 2, 10) });

// Performance Dashboard Metrics
var paginationRequestsCounter = Metrics.CreateCounter("pagination_requests_total", "Total de requisições de paginação", new[] { "page_number" });
var booksPerPageGauge = Metrics.CreateGauge("books_per_page", "Livros carregados por página", new[] { "page_number" });

await page.RouteAsync("**/api/books**", async route =>
{
    var requestStopwatch = Stopwatch.StartNew();
    var response = await route.FetchAsync();
    requestStopwatch.Stop();

    apiResponseTimeHistogram.WithLabels("/api/books").Observe(requestStopwatch.Elapsed.TotalSeconds);

    currentPage++;
    paginationRequestsCounter.WithLabels(currentPage.ToString()).Inc();

    if (response.Headers["content-type"]?.Contains("application/json") == true)
    {
        try
        {
            booksPerPageGauge.WithLabels(currentPage.ToString()).Set(booksResponse.Data.Count);
            // (...)
        }
        catch (Exception ex)
        {
            scrapingErrorsCounter.WithLabels("json_parse").Inc();
        }
    }

    await route.FulfillAsync(new RouteFulfillOptions { Response = response });
});
```

Github

Todos os exemplos estão disponíveis no repositório:

https://github.com/andrecustodio/DataES-20251206-ScrapingExamples

Perguntas?

Entre no grupo do WhatsApp !!!

VIX25

www.dataes.com.br