



# Mathias Kunto's blog

[About Mathias and this blog](#)[Webbshop](#)

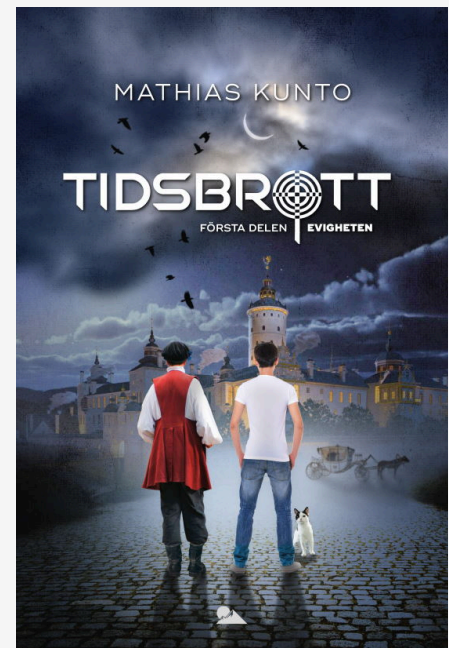
## Updated Exception enrichment functionality for added safety

Mathias Kunto October 13, 2025 EPiServer, Optimizely No Comments

Here is an improved version of the Exception enrichment functionality described in the article [Enriching your exceptions with information from Episerver](#). Updates has been made in order to make it more robust in regards to usage of abstract classes that may in some cases throw NotImplementedException, causing the enrichment code to terminate.

```
1 public static class ExceptionExtensions
2 {
3     private static readonly ILogger _logger =
4         LogManager.GetLogger(typeof(ExceptionExtensions));
5     private static readonly
6         Injected<HttpContextBase> _context;
7     private const int MaxValueLength =
8         150;
9     private const string Enriched =
10        "OS_Extra";
```

## Evigheten del 1: Tidsbrott



Första delen i serien [Evigheten](#), en historisk sci-fi för ungdomar. Provläs på [Smakprov.se](#) och köp ditt exemplar hos [Adlibris](#), [Bokus](#) eller [Akademibokhandeln](#). Du kan också låna den på ditt lokala bibliotek.

Tidsbrott finns som ljudbok hos bland andra [Storytel](#), [Nextory](#) och [Bookbeat](#) samt på många

[Cookies Settings](#)

By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.

Accept All Cookies

```

15     /// </summary>
16     /// <remarks>
17     /// Be really careful here so your
    code does not result in exceptions.
18     /// That would be _very_ bad for the
    website.
19     /// </remarks>
20     public static void Enrich(this
    Exception e)
21     {
22         if (e == null ||
    e.Data.Contains(Enriched))
23         {
24             return;
25         }
26
27         try
28         {
29             EnrichUnsafe(e);
30         }
31         catch (Exception ex)
32         {
33             _logger.Log(Level.Error,
    DoubleFaultMessage, ex);
34         }
35     }
36
37     private static void
    EnrichUnsafe(Exception e)
38     {
39         e.Data[Enriched] = string.Empty;
40
41         HttpContextBase context =
    _context.Service;
42
43         if (context != null)
44         {
45             UnsafeEnrichFrom(context.Request,
    e);
46
47             UnsafeEnrichFrom(context.User, e);
48
49             UnsafeEnrichFrom(context.Session,
    e);
50
51             UnsafeEnrichFrom(context.Request?.Browser,
    e);
52         }
53         else
54         {
55             e.Data["HttpContext"] = "null
    (possibly Optimizely scheduled job?);

```

vägen genom. Det finns inte ett kapitel, inte en sida då jag inte vill veta vad som händer härnäst.” — Linnea Lindblom, [Recension hos NSD](#), 2020-01-08

## Recent posts

[Updated Exception enrichment functionality for added safety](#)

[Allow only certain number of items in ContentAreas in Optimizely](#)

[Hiding and showing edit mode properties on same content type in Optimizely CMS depending on context](#)

[Changing Optimizely property return type yields MissingMethodException on getter](#)

[Object reference not set to an instance of an object in EPiServer.Shell.Web.Mvc.HttpMethod](#)

## Recent comments

By clicking “Accept All Cookies”, you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.

```

62     {
63         return !string.IsNullOrEmpty(value)
        && value.Length > maxLength
64         ? value.Substring(0, maxLength) +
        "... "
65         : value;
66     }
67
68     private static void
    UnsafeEnrichFrom(HttpRequestBase
    request, Exception e)
69     {
70         if (request == null)
71         {
72             return;
73         }
74
75         e.Data["Request_Url"] =
    GetValueOrDefault(() =>
    request.Url)?.ToString();
76         e.Data["Request_HttpMethod"] =
    GetValueOrDefault(() =>
    request.HttpMethod);
77         e.Data["Request_UserHostAddress"] =
    GetValueOrDefault(() =>
    request.UserHostAddress);
78         e.Data["Request_UserAgent"] =
    GetValueOrDefault(() =>
    request.UserAgent);
79         e.Data["Request_UrlReferrer"] =
    GetValueOrDefault(() =>
    request.UrlReferrer)?.ToString();
80         e.Data["Request_ContentType"] =
    GetValueOrDefault(() =>
    request.ContentType);
81         e.Data["Request_ContentLength"] =
    GetValueOrDefault(() =>
    request.ContentLength);
82         e.Data["Request_IsSecureConnection"]
    = GetValueOrDefault(() =>
    request.IsSecureConnection);
83
84         // Headers
85         var headers = new StringBuilder();
86         foreach (string key in
    request.Headers)
87         {
88             var value = request.Headers[key];
89             value = TruncateValue(value,
    MaxValueLength);
90             headers.AppendLine($" {key}:
    {value}");

```

[wałdis iljuczonok](#)  
[\(@tech\\_fellow\)](#) on React  
 and EPiServer: Aggregated  
 output with  
 EPiServer.ContentDeliveryApi

[Josef Ottosson](#) on React  
 and EPiServer – Custom  
 rendering of EPiServer  
 block in XhtmlString field  
 without partial block  
 controller

[Mathias Kunto](#) on React  
 and EPiServer – Custom  
 rendering of EPiServer  
 block in XhtmlString field  
 without partial block  
 controller

[Josef Ottosson](#) on React  
 and EPiServer – Custom  
 rendering of EPiServer  
 block in XhtmlString field  
 without partial block  
 controller

```

98
99     // Form (not logging actual values
    due to possible GDPR violation).
100     var form = new StringBuilder();
101     foreach (string key in request.Form)
102     {
103         var value = request.Form[key];
104         value = $"Length={value.Length}";
105         form.AppendLine($"    {key}:
    {value}");
106     }
107
108     if (form.Length > 0)
109     {
110         form.Insert(0,
    Environment.NewLine);
111         e.Data["Request_Form"] =
    form.ToString();
112     }
113
114     // Cookies
115     var cookies = new StringBuilder();
116     foreach (string key in
    request.Cookies)
117     {
118         var value =
    request.Cookies[key]?.Value;
119         value = TruncateValue(value,
    MaxValueLength);
120         cookies.AppendLine($"    {key}:
    {value}");
121     }
122
123     if (cookies.Length > 0)
124     {
125         cookies.Insert(0,
    Environment.NewLine);
126         e.Data["Request_Cookies"] =
    cookies.ToString();
127     }
128
129     // Files
130     var files = new StringBuilder();
131     foreach (string key in
    request.Files)
132     {
133         var file = request.Files[key];
134         if (file != null)
135         {
136             files.AppendLine($"    {key}:
    {file.FileName} ({file.ContentLength}
    bytes)");

```

By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.

```

files.ToString();
144     }
145 }
146
147 private static void
UnsafeEnrichFrom(IPrincipal principal,
Exception e)
148 {
149     if (principal?.Identity == null)
150     {
151         return;
152     }
153
154     if
(principal.Identity.IsAuthenticated)
155     {
156         e.Data["AuthenticatedUser"] = $"
{principal.Identity.Name}
(AuthenticationType:
{principal.Identity.AuthenticationType})";
157     }
158     else
159     {
160         e.Data["AuthenticatedUser"] =
$"User not authenticated.";
161     }
162 }
163
164 private static void
UnsafeEnrichFrom(HttpSessionStateBase
session, Exception e)
165 {
166     if (session == null)
167     {
168         return;
169     }
170
171     e.Data["Session_SessionID"] =
GetValueOrDefault(() =>
session.SessionID);
172
173     var sessionVars = new
StringBuilder();
174     foreach (string key in session)
175     {
176         var value =
session[key]?.ToString();
177         value = TruncateValue(value,
MaxValueLength);
178         sessionVars.AppendLine($" {key}:
{value}");
179     }

```

By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.

```

186     }
187 }
188
189 private static void
UnsafeEnrichFrom(HttpBrowserCapabilitiesBase
browser, Exception e)
190 {
191     if (browser == null)
192     {
193         return;
194     }
195
196     e.Data["Browser_Type"] =
GetValueOrDefault(() => browser.Type);
197     e.Data["Browser_Browser"] =
GetValueOrDefault(() =>
browser.Browser);
198     e.Data["Browser_Version"] =
GetValueOrDefault(() =>
browser.Version);
199     e.Data["Browser_Platform"] =
GetValueOrDefault(() =>
browser.Platform);
200     e.Data["Browser_MobileDeviceModel"]
= GetValueOrDefault(() =>
browser.MobileDeviceModel);
201 }
202
203 private static void
UnsafeEnrichWithOptimizelyData(Exception
e)
204 {
205     e.Data["StartPage_ID"] =
ContentReference.StartPage?.ID.ToString(CultureInfo.InvariantCulture)
?? string.Empty;
206
207     PageData page = null;
208     try
209     {
210         page =
ServiceLocator.Current.GetInstance<IPageRouteHelper>
()?.Page;
211     }
212     catch (Exception ex)
213     {
214         e.Data["CurrentPage"] =
$"Exception: {ex.Message}.";
215     }
216     if (page != null)
217     {
218         e.Data["CurrentPage"] = $"
{page.PageName} (ID:{page.PageLink}).";

```

By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.

```

225     }
226     catch (Exception ex)
227     {
228         e.Data["CurrentContent"] =
229         $"Exception: {ex.Message}.";
230     }
231     if (content != null)
232     {
233         e.Data["CurrentContent"] = $"
234         {content.Name} (ID:
235         {content.ContentLink}).";
236     }
237 }
238
239 /// <summary>
240 /// Used to safely get values from
241 properties.
242 /// </summary>
243 /// <typeparam name="T">Type of value
244 to get.</typeparam>
245 /// <param name="getterFunc">Func that
246 calls the getter. Example: () =>
247 request.HttpMethod</param>
248 /// <returns>The value to get, or the
249 type default if not able to get the
250 value.</returns>
251 /// <remarks>
252 /// When using abstract classes such
253 as HttpContextBase there are instances
254 when property getters
255 are not implemented (such as when
256 mocking requests for testing or running
257 scheduled jobs).
258 /// We need a safe way of getting
259 values that do not cause
260 NotImplementedException to be thrown.
261 /// </remarks>
262 private static T GetValueOrDefault<T>
263 (Func<T> getterFunc)
264 {
265     try
266     {
267         return getterFunc();
268     }
269     catch
270     {
271         return default;
272     }
273 }
274 }

```

By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.



Neoleaf AB in Stockholm.

Mathias Kunto's blog Copyright © 2025.

By clicking "Accept All Cookies", you agree to the storing of cookies on your device to enhance site navigation, analyze site usage, and assist in our marketing efforts.