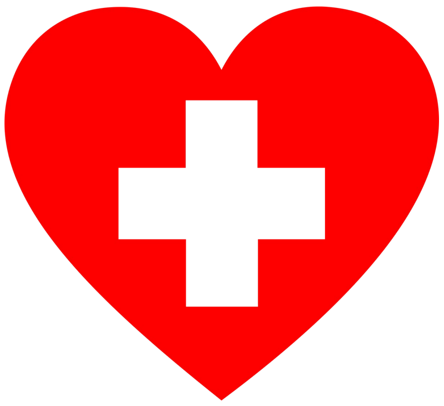
ALLAN THRAEN | 4 years ago | PDF |

Addon Development Optimizely (Episerver) C# Vision Demos & Prototypes

NEW PROJECT: OPTIMIZELY (EPISERVER) HEALTH CHECKER

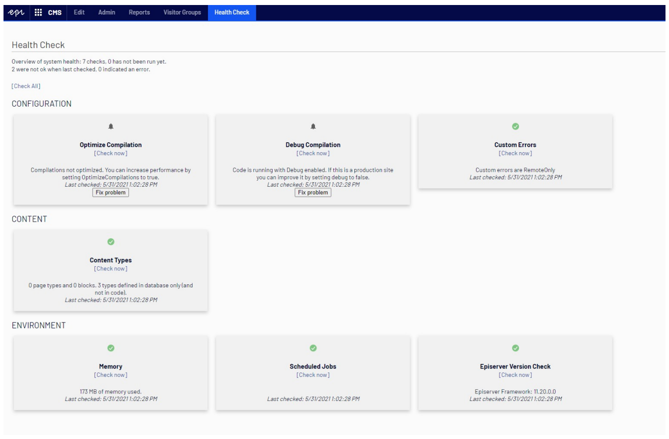


Wouldn't it be nice if there was just one place you could go and get an instant overview of the health of your Optimizely (Episerver) installation? With checks that are both technical, functional, business and covers product, addons and implementation? Here it is. Or at least the start of something that could be very useful.

This is an old dream of mine - and I'm happy to admit - inspired by solutions I've seen in several other cool pieces of software.

I started to build a health checker for Episerver a long time ago (back when it was called Episerver) - but I must admit that since I have a lot of project in the works, it can be hard to find enough time to dedicate to all of them. So, now I'm making it open source and hoping that this post might attract some dedicated collaborators that want to help out, as well as a lot of good ideas from Optimizely partners and customers along the way.

The project: Health Checker



Health Check

Overview of system health: 7 checks, 0 has not been run yet.
2 were not ok when last checked, 0 indicated an error.

[Check All]

CONFIGURATION

Optimize Compilation
[Check now]
Compilations not optimized. You can increase performance by setting OptimizeCompilations to true.
Last checked: 5/31/2021 1:02:28 PM
Fix problem

Debug Compilation
[Check now]
Code is running with Debug enabled. If this is a production site you can improve it by setting debug to false.
Last checked: 5/31/2021 1:02:28 PM
Fix problem

Custom Errors
[Check now]
Custom errors are RemoteOnly
Last checked: 5/31/2021 1:02:28 PM

CONTENT

Content Types
[Check now]
0 page types and 0 blocks. 3 types defined in database only (and not in code).
Last checked: 5/31/2021 1:02:28 PM

ENVIRONMENT

Memory
[Check now]
173 MB of memory used.
Last checked: 5/31/2021 1:02:28 PM

Scheduled Jobs
[Check now]
Last checked: 5/31/2021 1:02:28 PM

Episerver Version Check
[Check now]
Episerver Framework: 11.20.0.0
Last checked: 5/31/2021 1:02:28 PM

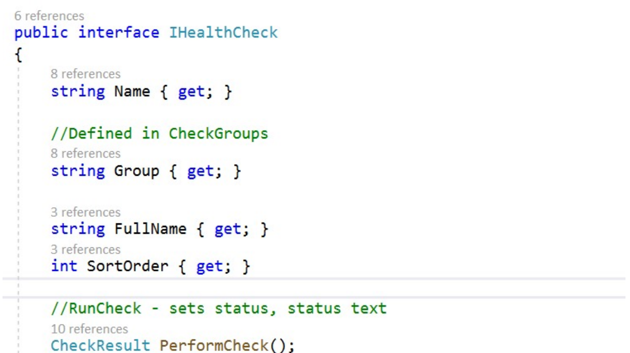
The concept is pretty simple: Allow developers to build a series of health checks for anything imaginable. Some could be included in the package along with the Health Checker, and others in their own packages depending on the Health Checker, and include a nice UI to give an overview of the status of each health check.

Whenever a Health check is performed, the status is stored along with a timestamp, so you can even see how old the check was - and if necessary perform it again. If all is OK in a health check, it's listed as green. If there is a problem, a red notification - and if there's something that could be optimized a neutral notification of that.

Some health checks will even allow an administrator to fix them right away by clicking a "Fix problem" button.

Health Checks in code

For now, all health checks simply needs to implement an IHealthCheck interface.

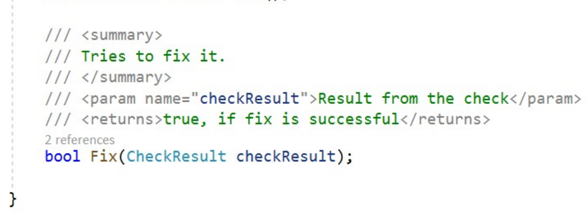


```
6 references
public interface IHealthCheck
{
    8 references
    string Name { get; }

    //Defined in CheckGroups
    8 references
    string Group { get; }

    3 references
    string FullName { get; }
    3 references
    int SortOrder { get; }

    //RunCheck - sets status, status text
    10 references
    CheckResult PerformCheck();
```

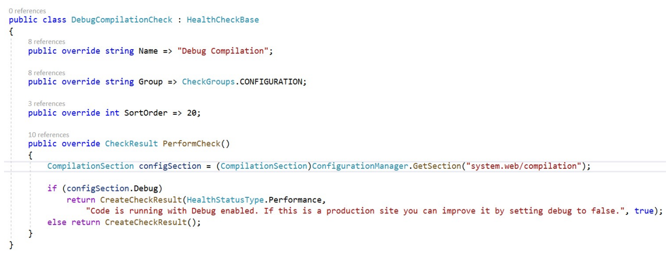


```
    /// <summary>
    /// Tries to fix it.
    /// </summary>
    /// <param name="checkResult">Result from the check</param>
    /// <returns>true, if fix is successful</returns>
    2 references
    bool Fix(CheckResult checkResult);

}
```

As you can see you can even implement a method to try to fix a given scenario.

To make it easier to implement a lot of health checks, there's also a base class that implements the interface. Here is an example of a simple health check to see if the site is running in debug mode:



```
0 references
public class DebugCompilationCheck : HealthCheckBase
{
    8 references
    public override string Name => "Debug Compilation";

    8 references
    public override string Group => CheckGroups.CONFIGURATION;

    3 references
    public override int SortOrder => 20;

    10 references
    public override CheckResult PerformCheck()
    {
        CompilationSection configSection = (CompilationSection)ConfigurationManager.GetSection("system.web/compilation");

        if (configSection.Debug)
            return CreateCheckResult(HealthStatusType.Performance,
                "Code is running with Debug enabled. If this is a production site you can improve it by setting debug to false.", true);
        else return CreateCheckResult();
    }
}
```

Health checks

When it comes to which things we can check, the limit goes at our imagination.

So far, I've been implementing a handful pretty basic checks - but there is already a long list of ideas - and I hope it will grow even longer with your suggestions.

How about:

- Security checks to identify typical vulnerabilities on an Episerver site
- Content type redundancy checks - when a site has a bit of history there are often way too many content types and they often redundant.
- Content Approval checks - to ensure that there is not a lot of content waiting on an approver that has left the company?
- Geolocation - that lets you know if your current geolocation database is old?
- SEO checkers that look for the most typical SEO problems?
- Email settings validator to see if your config settings for email works as it should?
- And much, much more.

Project status

"Where can I get the Nuget package for this amazing technology?" I hear you cry. Well - it's not ready yet. This is merely a post to invite collaborators to want to join the project - or anybody else who has ideas for health checks or other functionality.

The project can be found on GitHub currently: **https://github.com/CodeArtDK/CodeArt.Episerver.Health**.

I invite anyone willing to contribute to leave a comment or send me a message - or perhaps just leave an idea as an issue on GitHub. But in particular I would love to include someone on the team with these skills:

- Episerver Addon experience
- Frontend skills
- Detailed knowledge of popular Episerver addons that would be good to have health checks for
- Commerce expert
- DXP expert to introduce checks for the DXP environments
- Security expertise.

Addon Development | Optimizely (Episerver) | C# | Vision Demos & Prototypes

CodeArt ApS
Teknikerbyen 5, 2830 Virum, Denmark
Email: info@codeart.dk
Phone: +45 26 13 66 96
CVR: 39680688