

ZABBIX

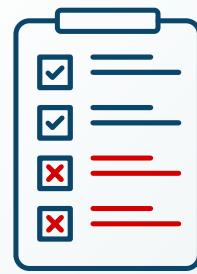
Browser Monitoring in Zabbix 7.0

Kaspars Mednis

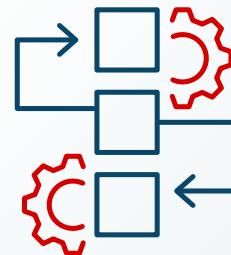
Training project manager

Website monitoring requirements

- ▶ Regular Testing: Perform regular tests to ensure continuous monitoring
- ▶ Automation: Automate monitoring tasks to reduce manual effort
- ▶ Charts and Graphs: Visual representations of monitoring data and trends.
- ▶ Screenshots: Visual examples of issues



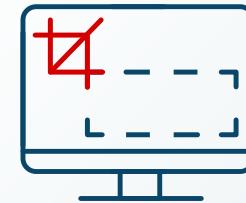
Testing



Automation



Graphs



Screenshots

Synthetic web monitoring

Synthetic Monitoring involves simulating **user interactions** with a website using automated scripts to test performance and functionality

Key Characteristics:

- ▶ **Proactive Testing:** Performed regularly, even if there are no real users on the site.
- ▶ **Scenario-Based Testing:** Can test specific scenarios, such as login processes, form submissions, or transaction flows.
- ▶ **Baselines and Benchmarks:** Helps in establishing performance baselines and benchmarks for comparison over time.

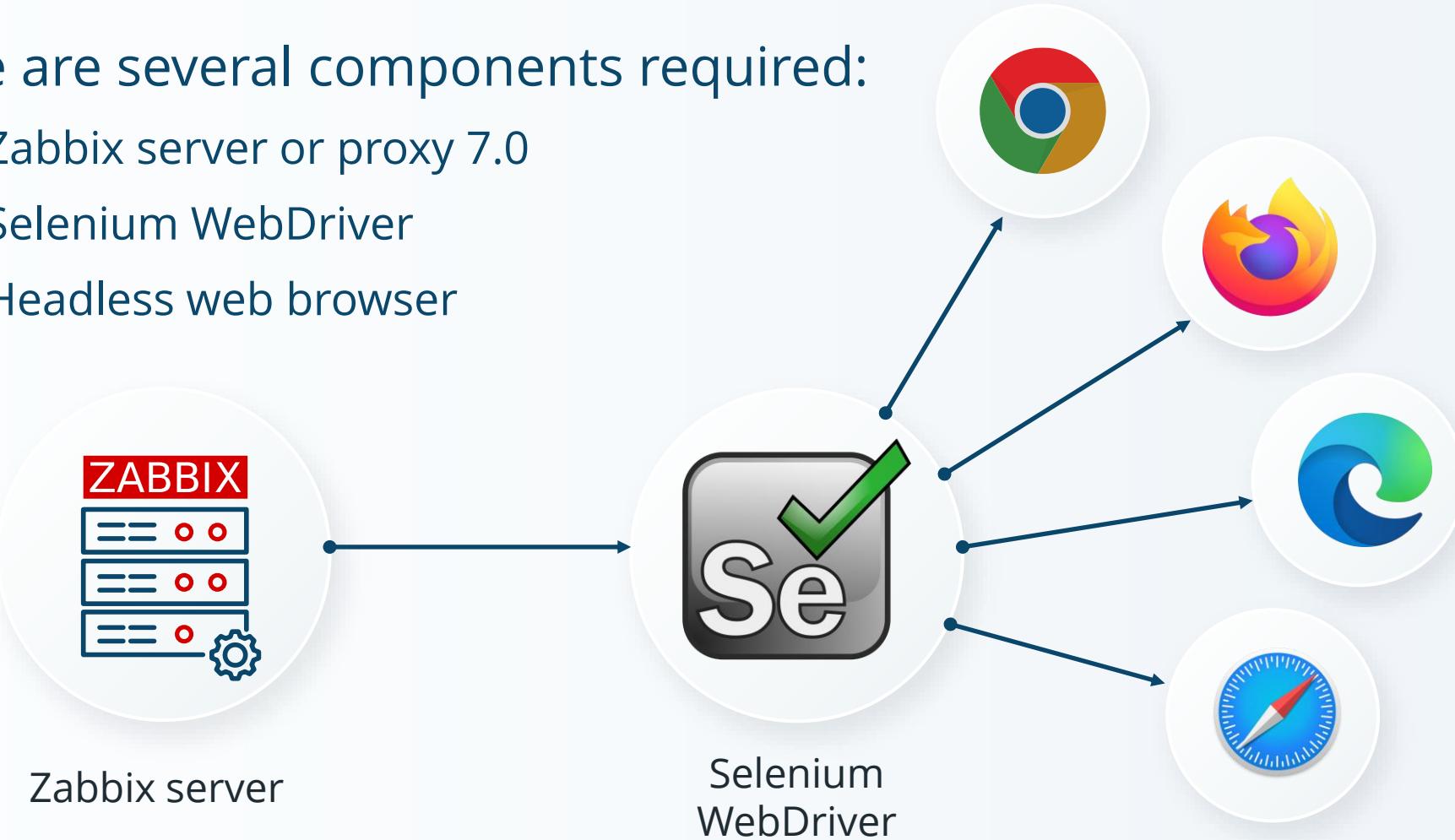
Monitoring Environment Setup



Monitoring requirements

There are several components required:

- ▶ Zabbix server or proxy 7.0
- ▶ Selenium WebDriver
- ▶ Headless web browser



Zabbix server configuration

```
##### Browser monitoring #####
### Option: WebDriverURL
#      WebDriver interface HTTP[S] URL. For example http://localhost:4444 used with
#      Selenium WebDriver standalone server.
#
# Mandatory: no
# Default:
# WebDriverURL=
WebDriverURL=http://192.168.0.1:4444

### Option: StartBrowserPollers
#      Number of pre-forked instances of browser item pollers.
#
# Mandatory: no
# Range: 0-1000
# Default:
# StartBrowserPollers=1
StartBrowserPollers=3
```

Example setup with containers

```
[Unit]
Description=Zabbix Server

[Container]
ContainerName=zabbix-server
Image=docker.io/zabbix/zabbix-server-mysql:alpine-trunk
Network=training.network
PublishPort=10051:10051
Environment=DB_SERVER_HOST=mysql
...
Environment=ZBX_WEBDRIVERURL=http://selenium:4444/wd/hub
Environment=ZBX_STARTBROWSERPOLLERS=3

[Service]
Restart=always

[Install]
WantedBy=default.target

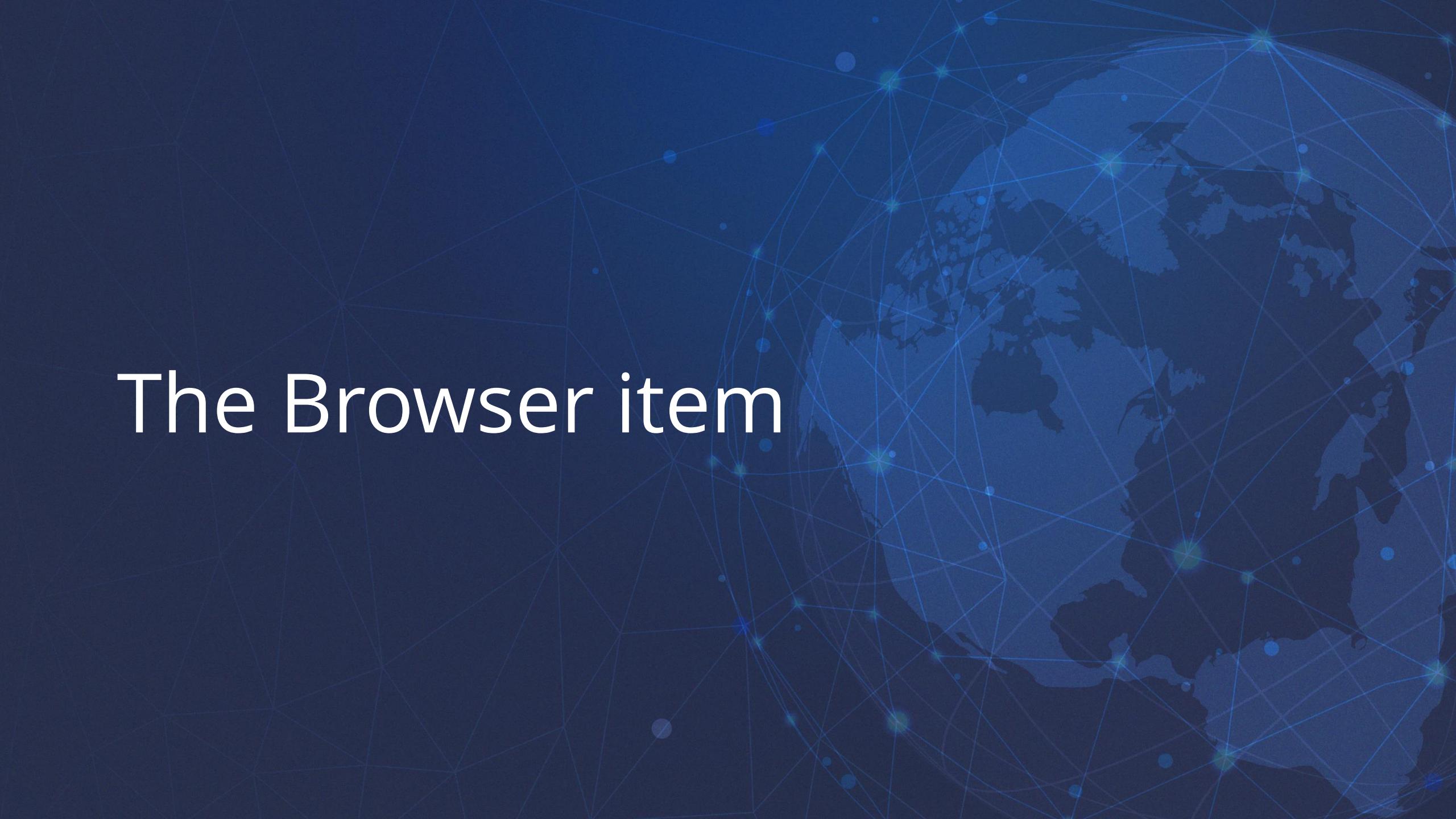
[Unit]
Description=Selenium with chrome

[Container]
ContainerName=selenium
Image=docker.io/selenium/standalone-chrome:latest
Network=training.network
PublishPort=4444:4444

[Service]
Restart=always

[Install]
WantedBy=default.target
```

The Browser item



Browser item

Zabbix 7.0 introduces new item type: Browser

New item

Item Tags Preprocessing

* Name Zabbix website

Type Browser

* Key website.get.data

Type of information Text

Parameters

Name	Value
<input type="text"/>	<input type="button"/>

Add

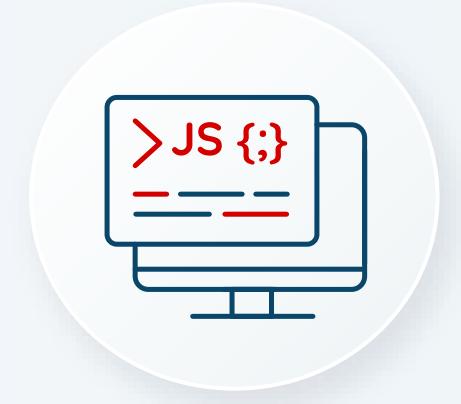
* Script

```
var browser = new Browser(Browser.chromeOptions());
```

* Update interval 1m

65346 characters remaining

Apply Cancel



JavaScript

```
1 var browser = new Browser(Browser.chromeOptions());  
2  
3 try {  
4     browser.navigate("http://example.com");  
5     browser.collectPerfEntries();  
6 }  
7 finally {  
8     return JSON.stringify(browser.getResult());  
9 }
```

Browser item timeout

Data collection timeout can be specified for the browser item:

- ▶ On the Zabbix server level
- ▶ On the Zabbix proxy level
- ▶ On the individual item level

Item override

* Timeout	Global	Override	3m	Timeouts
* History	Do not store	Store up to	31d	

Timeouts for item types	
* Zabbix agent	3s
* Simple check	3s
* SNMP agent	3s
* External check	3s
* Database monitor	3s
* HTTP agent	3s
* SSH agent	3s
* TELNET agent	3s
* Script	3s
* Browser	1m

Browser item output

The browser item collects all performance metrics in the JSON format

```
{  
    "duration": 5.4627423286438,  
    "performance_data": {  
        "details": [  
            {  
                "navigation": {  
                    "activation_start": 0,  
                    "connect_end": 0.0639000000059605,  
                    "connect_start": 0.022,  
                    "critical_ch_restart": 0,  
                    "decoded_body_size": 202169,  
                    "delivery_type": "",  
                    "dom_complete": 5.30179999999702,  
                    "dom_content_loaded_event_end": 3.6295,  
                    "dom_content_loaded_event_start": 3.53090000000596,  
                    "domain_lookup_end": 0.022,  
                    "domain_lookup_start": 0.0219000000059605,  
                    .....  
                }  
            }  
        ]  
    }  
}
```



Browser item parameters

It is possible to send custom parameters to the JavaScript:

- ▶ Write name and value pairs in the Parameters
- ▶ User macros can be used as the browser item parameters

Parameters	Name	Value	Action
	browser	{\$WEBSITE.BROWSER}	Remove
	domain	{\$WEBSITE.DOMAIN}	Remove
	height	{\$WEBSITE.SCREEN.HEIGHT}	Remove
	path	{\$WEBSITE.PATH}	Remove
	scheme	{\$WEBSITE.SCHEME}	Remove
	width	{\$WEBSITE.SCREEN.WIDTH}	Remove
	...		

```
const browser = new Browser(Website.getOptions(Website.params.browser));
```

Individual metrics

Data are extracted from the browser item using dependent items:

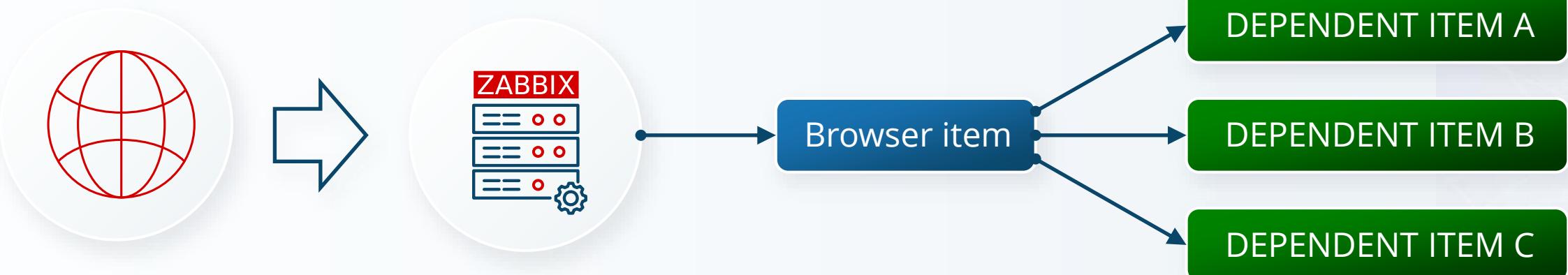
- ▶ Browser item collects data in JSON format
- ▶ Dependent items use the JSONPath preprocessing step to extract data

Preprocessing steps ?		Name	Parameters
1:	JSONPath	\$.performance_data.summary.navigation.dns_lookup_time	
2:	Custom multiplier	0.001	

[Add](#)

Browser item and dependent items

Name ▾	Triggers	Key	Interval	History	Trends	Type	Status
Website Get data		website.get.data	5m	0		Browser	Enabled
Website Get data: Navigation response time		website.navigation.response_time	31d	0		Dependent item	Enabled
Website Get data: Navigation request time		website.navigation.request_time	31d	0		Dependent item	Enabled
Website Get data: Navigation encodedBody size		website.navigation.encoded_size	31d	0		Dependent item	Enabled
Website Get data: Navigation domContentLoaded time		website.navigation.dom_content_loaded_time	31d	0		Dependent item	Enabled
Website Get data: Navigation DNS lookup time		website.navigation.dns_lookup_time	31d	0		Dependent item	Enabled



Monitoring scenarios

Advanced scenarios

Monitoring scenarios are created in JavaScript (Duktape engine)

Because browser item emulates a real browser, it is possible to:

- ▶ Log on and log out from the website
- ▶ Fill and submit different forms
- ▶ Navigate through multiple pages
- ▶ Simulate a click on the webpage
- ▶ Create complex if - then scenarios

Homepage monitoring

Scenario: Simulate a user opening the website's homepage

Steps:

- ▶ Navigate to the homepage URL
- ▶ Measure the time it takes for the page to fully load
- ▶ Check for any errors or missing elements

Purpose: Ensure the homepage loads quickly and correctly

Navigation Flow Test

Scenario: Simulate a user navigating through multiple pages

Steps:

- ▶ Navigate to the homepage
- ▶ Click on a main menu link to go to a secondary page
- ▶ From the secondary page, navigate to another linked page
- ▶ Return to the homepage using the site's navigation

Purpose: Verify that navigation links work correctly

Login functionality test

Scenario: Simulate a user logging into the website

Steps:

- ▶ Navigate to the login page
- ▶ Enter a valid username and password
- ▶ Click the login button
- ▶ Verify successful login by checking for a specific element on the page

Purpose: Confirm that the login process is functional and secure.

Search Functionality Test

Scenario: Simulate a user performing a search on the website.

Steps:

- ▶ Navigate to the search page
- ▶ Enter a search query into the search bar
- ▶ Click the search button
- ▶ Verify that search results are displayed and relevant to the query

Purpose: Ensure the search feature works correctly

Shopping Cart and Checkout Test

Scenario: Simulate a user adding items to the shopping cart and completing a purchase

Steps:

- ▶ Navigate to a product page
- ▶ Add the product to the shopping cart
- ▶ Proceed to the checkout page and enter payment and shipping information
- ▶ Complete the purchase
- ▶ Verify order confirmation and receipt

Purpose: Ensure the shopping and checkout process are functional

Screenshots



Taking screenshots

The browser item can take screenshot from the monitored pages:

- ▶ Screenshot is included in the JSON object in base64 format
- ▶ It is extracted into a binary dependent item

Item

Item Tags 1 Preprocessing 1

* Name: Website Screenshot

Type: Dependent item

* Key: website.screenshot

Select

Type of information: Binary

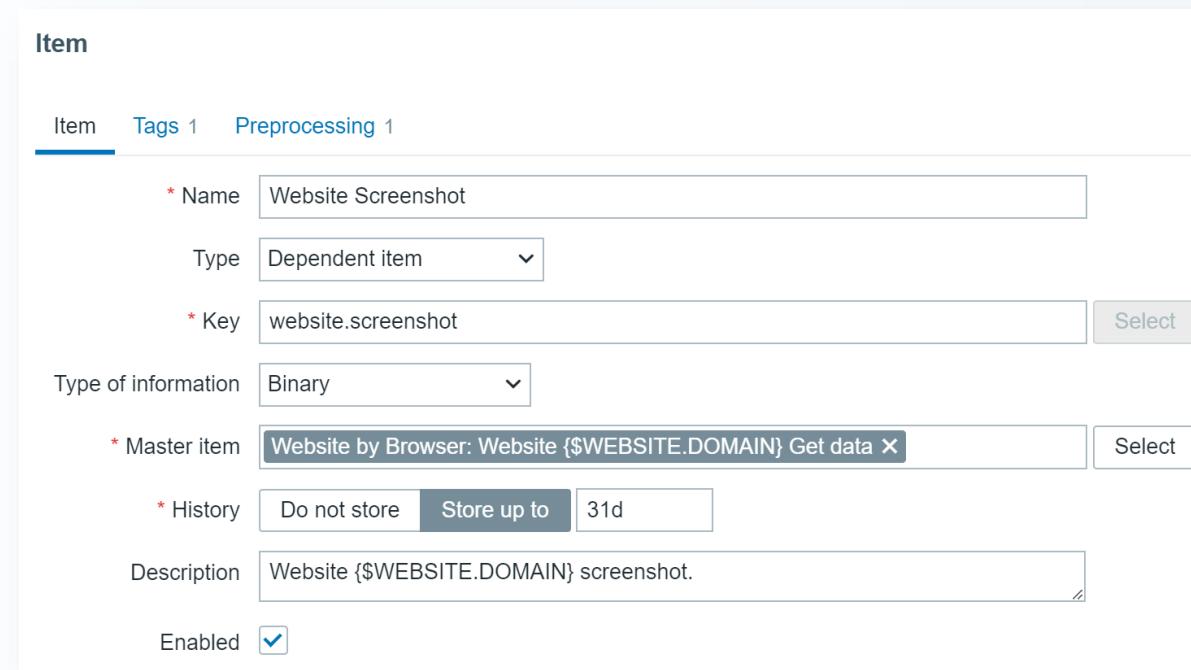
* Master item: Website by Browser: Website {\$WEBSITE.DOMAIN} Get data

Select

* History: Do not store Store up to: 31d

Description: Website {\$WEBSITE.DOMAIN} screenshot.

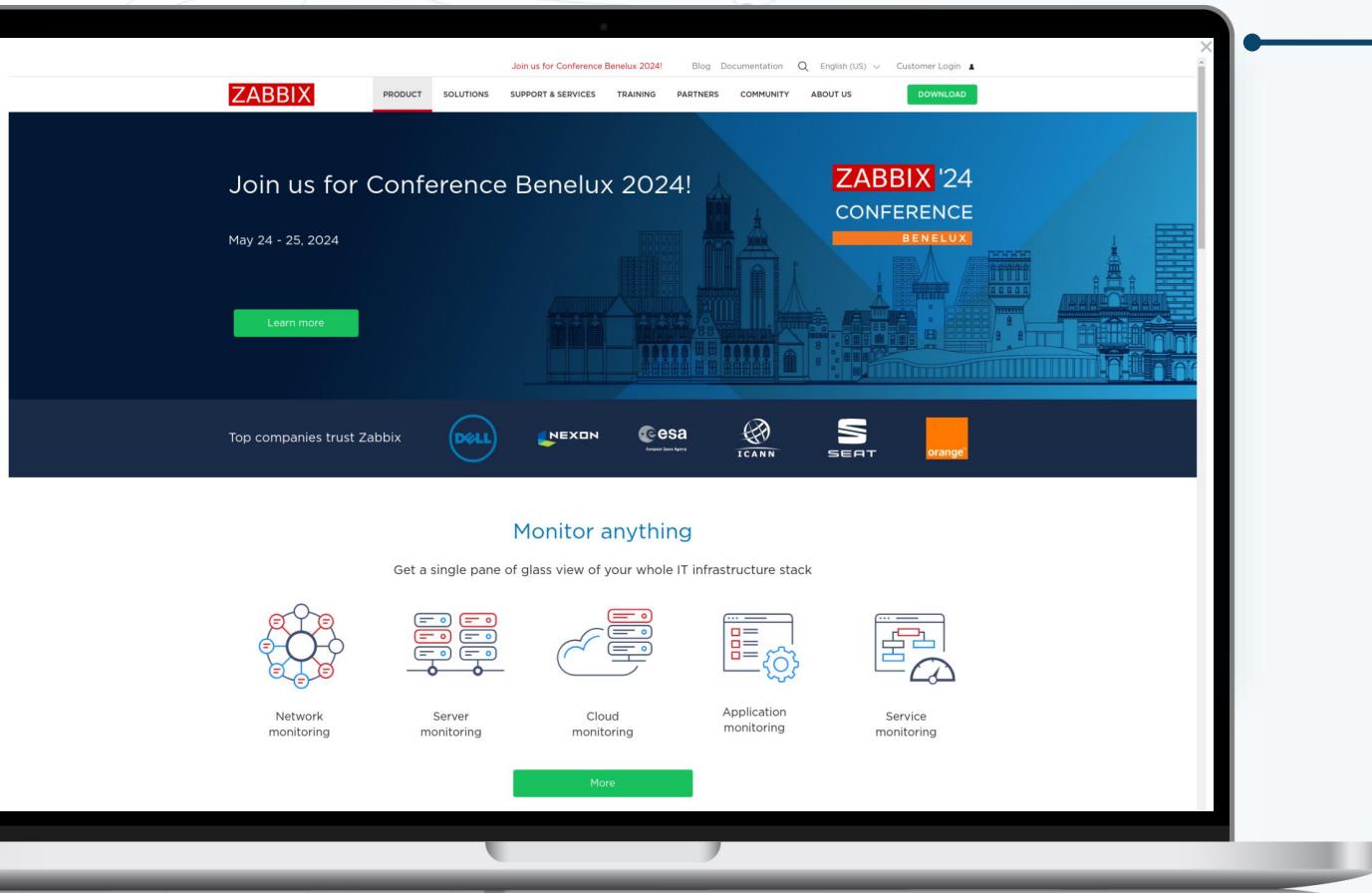
Enabled:



Screenshot details

Zabbix supports up to 8K x 8K screenshots

- ▶ Default size is 1920 x 1080
- ▶ Screenshot can be displayed using the "Item history" widget
- ▶ Screenshot size is specified by the `browser.setScreenSize(x,y)` method



Item history widget example

Timestamp	Screenshot
2024-05-22 01:46:28 PM	
2024-05-22 01:44:28 PM	
2024-05-22 01:43:26 PM	
2024-05-22 01:42:28 PM	
2024-05-22 01:41:28 PM	
2024-05-22 01:40:28 PM	
2024-05-22 01:39:29 PM	
2024-05-22 01:38:28 PM	

The screenshot shows the official Zabbix website homepage. At the top, there's a navigation bar with links for 'PRODUCT', 'SOLUTIONS', 'SUPPORT & SERVICES', 'TRAINING', 'PARTNERS', 'COMMUNITY', 'ABOUT US', and a 'DOWNLOAD' button. A search bar and language selection ('English (US)') are also present. The main header features the text 'Join us for Conference Benelux 2024!' and 'ZABBIX '24 CONFERENCE BENELUX'. Below this, there's a section titled 'Top companies trust Zabbix' with logos for Dell, NEXON, esa, ICANN, SEAT, and orange. The central part of the page has a large banner with the text 'Monitor anything' and 'Get a single pane of glass view of your whole IT infrastructure stack'. It includes icons for Network monitoring, Server monitoring, Cloud monitoring, Application monitoring, and Service monitoring, each with a brief description below it. A 'More' button is located at the bottom of this section.

Out-of-box monitoring

Website by Browser template

Zabbix 7.0 comes with the "Website by Browser" template

Template

Template Tags 2 Macros 9 Value mapping

* Template name Website by Browser

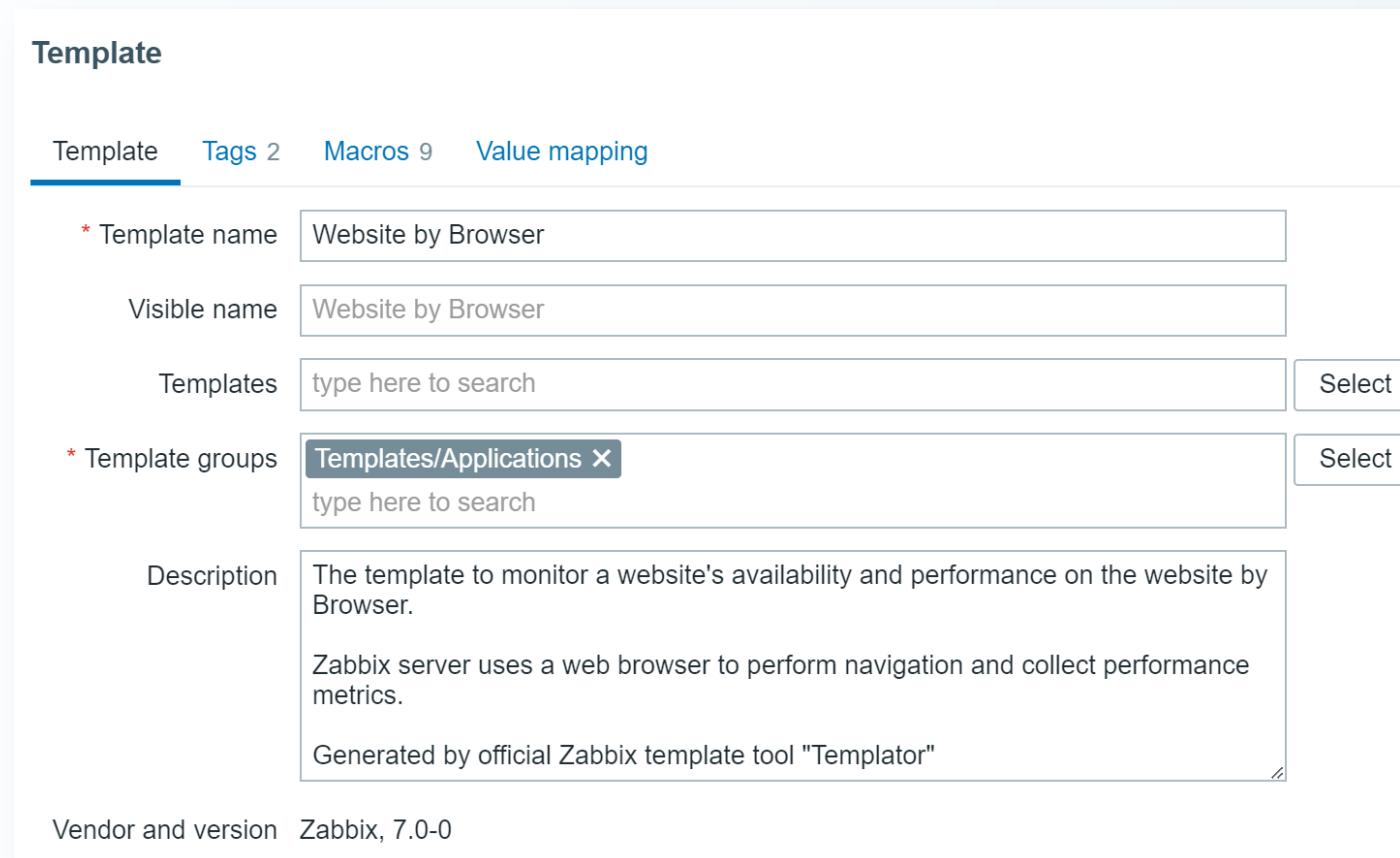
Visible name Website by Browser

Templates type here to search Select

* Template groups Templates/Applications × Select
type here to search

Description The template to monitor a website's availability and performance on the website by Browser.
Zabbix server uses a web browser to perform navigation and collect performance metrics.
Generated by official Zabbix template tool "Templator"

Vendor and version Zabbix, 7.0-0



Template content

The new template includes:

- ▶ A "Browser" item with a data collection script
- ▶ 26 dependent items for individual metrics
- ▶ 3 predefined triggers
 - Failed to get metrics data
 - Website navigation load event time is too slow
 - Website resource load event time is too slow
- ▶ 9 User macros (browser type, site name, screenshot dimensions, etc)
- ▶ 2 predefined graphs
- ▶ A host dashboard

Specifying monitoring parameters

Macro	Value	Description	
{\$WEBSITE.BROWSER}	chrome	Browser to be used for data collection.	Remove
{\$WEBSITE.DOMAIN}	www.zabbix.com	The domain name.	Remove
{\$WEBSITE.GET.DATA.INTERVAL}	10m	Update interval for get raw data item.	Remove
{\$WEBSITE.NAVIGATION.LOAD.WARN}	5	The maximum browser response time expressed in seconds for a trigger expression.	Remove
{\$WEBSITE.PATH}	value	The path to resource.	Remove
{\$WEBSITE.RESOURCE.MAX.WARN}	5	The maximum browser response time expressed in seconds for a trigger expression.	Remove
{\$WEBSITE.SCHEME}	https	The request scheme, which may be either HTTP or HTTPS.	Remove
{\$WEBSITE.SCREEN.HEIGHT}	1080	Screen size height in pixels, used for screenshot.	Remove
{\$WEBSITE.SCREEN.WIDTH}	1920	Screen size width in pixels, used for screenshot.	Remove

Predefined browsing script

JavaScript

```
26  getPerformance() {
27      const browser = new Browser(Website.getOptions(Website.params.browser));
28      const url = Website.params.scheme + '://' + Website.params.domain + '/' + Website.params.path
29      const screenshot = '';
30      browser.setScreenSize(Number(Website.params.width), Number(Website.params.height))
31      browser.navigate(url);
32      browser.collectPerfEntries();
33      screenshot = browser.getScreenshot();
34      const result = browser.getResult();
35      result.screenshot = screenshot;
36
37      return JSON.stringify(result);
38  }
39}
40
41 try {
42     Website.setParams(JSON.parse(value));
43     return Website.getPerformance();
44
45 } catch (error) {
46     error += (String(error).endsWith('.')) ? '' : '.';
47     Zabbix.log(3, '[ Website get metrics] ERROR: ' + error);
48     return JSON.stringify({ 'error': error });

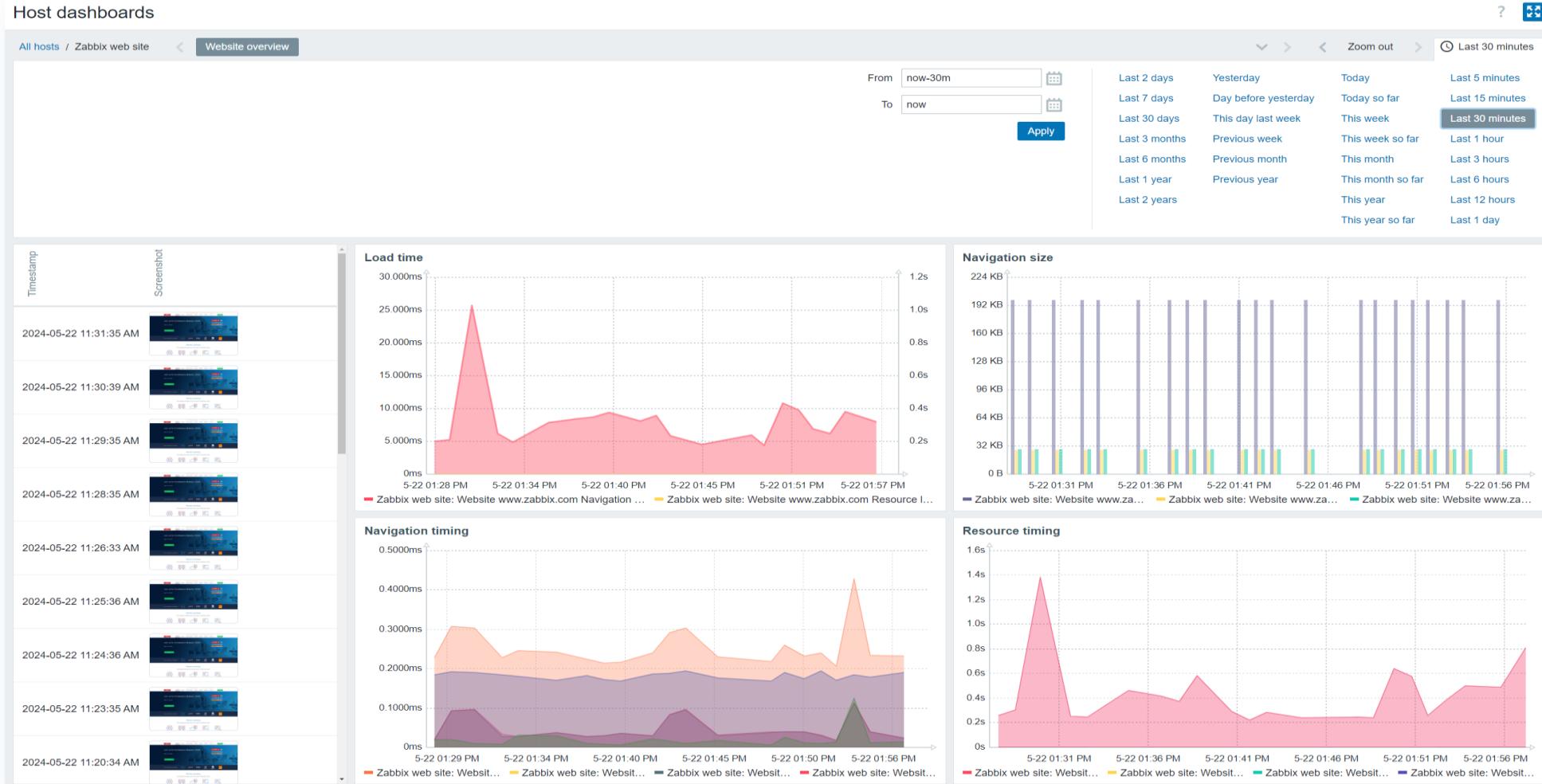
```

63959 characters remaining

Apply

Cancel

Result



ZABBIX

Thank you

Kaspars Mednis

Training project manager