PebbleTemplates / **pebble** Public

Code Issues 25 Pull requests 2 Discussions Actions Security and

Commit b3451c8

ebussieres authored on Dec 11, 2025 · 3 / 3 · Verified

```
CVE-2025-1686 (#715)
* fix: CVE-2025-1686

- use only a ClasspathLoader by default
- Modify the `FileLoader` to use a mandatory sandboxed base directory parameter

* chore: update version

* feat: protect FileLoader against path traversal

* chore: fix unit test to run on Windows

* feat: simplify FileLoader and add some unit tests

* Update FileLoader.java

* Update FileLoader.java
```

master (#715) · v4.1.1 ••• 4.1.0

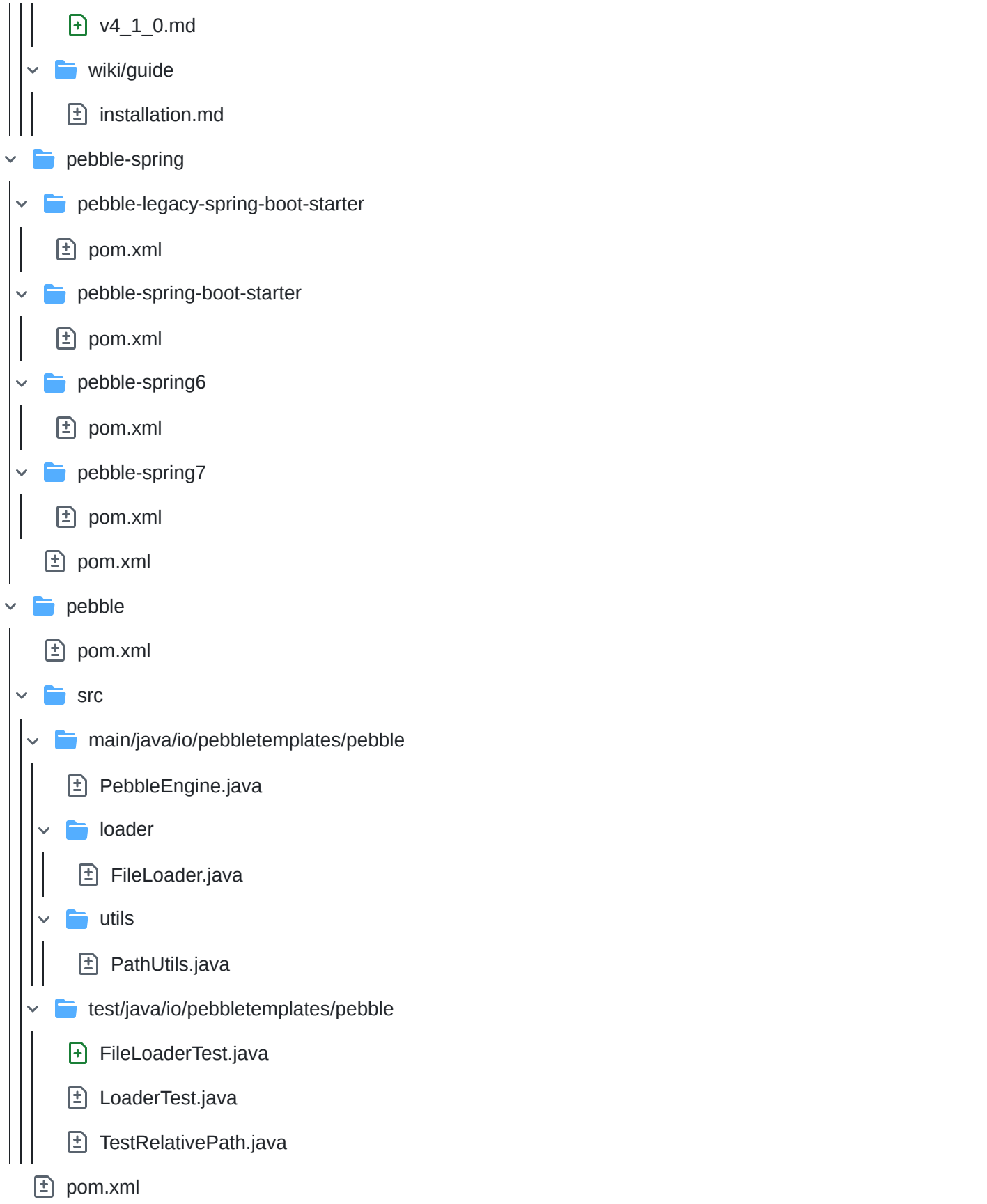
1 parent a5a38be commit b3451c8

18 files changed +226 -152 lines changed

Top

Filter files…

- README.md
- docs
  - pom.xml
  - src/orchid/resources
    - changelog
      - v4_0_1.md



18 files changed +226 -152 lines changed

Search within code

README.md

```
@@ -6,6 +6,12 @@ includes integrated support for internationalization.
6 6
7 7  For more information please visit the [website](https://pebbletemplates.io).
8 8
9 +  # Breaking changes in version 4.1.x
10 +
11 +  - If you do not provide a custom Loader, Pebble will now use only a `ClasspathLoader` by default, same as the spring autoconfiguration.
12 +    Before that, it would have used an instance of the `DelegatingLoader` which consists of a `ClasspathLoader` and a `FileLoader` behind the scenes to find your templates.
13 +  - Modify the `FileLoader` to use a mandatory sandboxed base directory parameter.
14 +
9 15  # Breaking changes in version 4.0.x
10 16
11 17  - Use one of the following artifactId according to the spring boot version that you are using
```

docs/pom.xml

```
@@ -5,7 +5,7 @@
5 5    <parent>
6 6      <groupId>io.pebbletemplates</groupId>
7 7      <artifactId>pebble-project</artifactId>
8   -    <version>4.0.1-SNAPSHOT</version>
  8 +    <version>4.1.0-SNAPSHOT</version>
9 9    </parent>
10 10
11 11   <artifactId>docs</artifactId>
```

docs/src/orchid/resources/changelog/v4_0_1.md

Load Diff

This file was deleted.

docs/src/orchid/resources/changelog/v4_1_0.md

```
@@ -0,0 +1,17 @@
+ ---
+ version: '4.1.0'
+ ---
+
+ # BREAKING CHANGES
+ - Modify the `FileLoader` to use a mandatory sandboxed base directory parameter. (#715)
+ - If you do not provide a custom Loader, Pebble will now use only a `ClasspathLoader` by default, same as the spring autoconfiguration. (#715)
+   Before that, it would have used an instance of the `DelegatingLoader` which consists of a `ClasspathLoader` and a `FileLoader` behind the scenes to find your templates.
+
+ # New Features
+ - Use a default existing format of `yyyy-MM-dd'T'HH:mm:ssZ` when using date filter with a string (#677)
+ - Look for exact method / field match when doing reflection. Look for method get/is/has if none match
+ - Update some dependencies (#709)
+
+ # Bug Fixes
+ - NaN must return false instead of throwing an exception (#695)
+ - [CVE-2025-1686](https://nvd.nist.gov/vuln/detail/CVE-2025-1686).
```

...orchid/resources/wiki/guide/installation.md

```
@@ -61,18 +61,17 @@ finding your templates.

 Pebble ships with the following loader implementations:

+ - `DelegatingLoader`: Delegates responsibility to a collection of children loaders.
   - `ClasspathLoader`: Uses a classloader to search the current classpath.
- - `FileLoader`:  Finds templates using a filesystem path.
+ - `FileLoader`:  Finds templates using a filesystem path. Must provide a mandatory absolute base path.
```

```
66  67    - `ServletLoader`:  Uses a servlet context to find the template. This is the recommended loader for use within an
67  68    application server but is not enabled by default.
68  69    - `Servlet5Loader`:  Same as `ServletLoader`, but for Jakarta Servlet 5.0 or newer.
69      - - `StringLoader`: Considers the name of the template to be the contents of the template.
70      - - `DelegatingLoader`: Delegates responsibility to a collection of children loaders.
71  70    - `MemoryLoader`: Loader that supports inheritance and doesn't require a filesystem. This is useful for applications
        71  + - `StringLoader`: Considers the name of the template to be the contents of the template. Should not be used in a production environment. It is primarily for testing and debugging. Many tags may not work when using this loader, such as "extends", "imports", etc.
72  72      that retrieve templates from a database for example.
73  73
74      - If you do not provide a custom Loader, Pebble will use an instance of the `DelegatingLoader` by default.
75      - This delegating loader will use a `ClasspathLoader` and a `FileLoader` behind the scenes to find your templates.
        74  + If you do not provide a custom Loader, Pebble will use an instance of the `ClasspathLoader` by default.
76  75
77  76    ## Pebble Engine Settings
78  77
        @@ -85,7 +84,7 @@ All the settings are set during the construction of the `PebbleEngine` object.
85  84    | `tagCache` | An implementation of a ConcurrentMap cache that the Pebble engine will use for {{ anchor('cache tag', 'cache') }}. | Default implementation is `ConcurrentMapTagCache` and another implementation based on Caffeine is available (`CaffeineTagCache`) |
86  85    | `defaultLocale` | The default locale which will be passed to each compiled template. The templates then use this locale for functions such as i18n, etc. A template can also be given a unique locale during evaluation.  | `Locale.getDefault()` |
87  86    | `executorService` | An `ExecutorService` that allows the usage of some advanced multithreading features, such as the `parallel` tag. | `null` |
88      - | `loader` | An implementation of the `Loader` interface which is used to find templates. | An implementation of the `DelegatingLoader` which uses a
```

```
   `ClasspathLoader` and a `FileLoader` behind the scenes. |
87 + | `loader` | An implementation of the `Loader` interface which is used to find templates. | An implementation of the `ClasspathLoader` |
89 88   | `strictVariables` | If set to true, Pebble will throw an exception if you try to access a variable or attribute that does not exist (or an attribute of a null variable). If set to false, your template will treat non-existing variables/attributes as null without ever skipping a beat. | `false` |
90 89   | `methodAccessValidator` | Pebble provides two implementations. NoOpMethodAccessValidator which do nothing and BlacklistMethodAccessValidator which checks that the method being called is not blacklisted. | `BlacklistMethodAccessValidator`
91 90   | `literalDecimalTreatedAsInteger` | option for treating literal decimals as `int`. Otherwise it is `long`. | `false` |
```

...g/pebble-legacy-spring-boot-starter/pom.xml

```
   @@ -4,7 +4,7 @@
4  4     <parent>
5  5       <artifactId>pebble-spring</artifactId>
6  6       <groupId>io.pebbletemplates</groupId>
7    -     <version>4.0.1-SNAPSHOT</version>
   7   +     <version>4.1.0-SNAPSHOT</version>
8  8     </parent>
9  9
10 10    <artifactId>pebble-legacy-spring-boot-starter</artifactId>
```

...e-spring/pebble-spring-boot-starter/pom.xml

```
   @@ -4,7 +4,7 @@
4  4     <parent>
5  5       <artifactId>pebble-spring</artifactId>
6  6       <groupId>io.pebbletemplates</groupId>
7    -     <version>4.0.1-SNAPSHOT</version>
   7   +     <version>4.1.0-SNAPSHOT</version>
8  8     </parent>
9  9
10 10    <artifactId>pebble-spring-boot-starter</artifactId>
```

pebble-spring/pebble-spring6/pom.xml

```
@@ -4,7 +4,7 @@
    <parent>
      <artifactId>pebble-spring</artifactId>
      <groupId>io.pebbletemplates</groupId>
-     <version>4.0.1-SNAPSHOT</version>
+     <version>4.1.0-SNAPSHOT</version>
    </parent>

    <artifactId>pebble-spring6</artifactId>
```

pebble-spring/pebble-spring7/pom.xml

```
@@ -4,7 +4,7 @@
    <parent>
      <artifactId>pebble-spring</artifactId>
      <groupId>io.pebbletemplates</groupId>
-     <version>4.0.1-SNAPSHOT</version>
+     <version>4.1.0-SNAPSHOT</version>
    </parent>

    <artifactId>pebble-spring7</artifactId>
```

pebble-spring/pom.xml

```
@@ -4,7 +4,7 @@
    <parent>
      <groupId>io.pebbletemplates</groupId>
      <artifactId>pebble-project</artifactId>
-     <version>4.0.1-SNAPSHOT</version>
+     <version>4.1.0-SNAPSHOT</version>
    </parent>

    <artifactId>pebble-spring</artifactId>
```

pebble/pom.xml

```
@@ -3,7 +3,7 @@
3  3    <parent>
4  4      <groupId>io.pebbletemplates</groupId>
5  5      <artifactId>pebble-project</artifactId>
6     -    <version>4.0.1-SNAPSHOT</version>
   6  +    <version>4.1.0-SNAPSHOT</version>
7  7    </parent>
8  8
9  9    <artifactId>pebble</artifactId>
```

...io/pebbletemplates/pebble/PebbleEngine.java

```
@@ -9,45 +9,40 @@
9  9  package io.pebbletemplates.pebble;
10 10
11 11
   12 + import
   io.pebbletemplates.pebble.attributes.methodaccess.BlacklistMethodAccessValidato
   r;
   13 + import io.pebbletemplates.pebble.attributes.methodaccess.MethodAccessValidator;
12 14  import io.pebbletemplates.pebble.cache.CacheKey;
13 15  import io.pebbletemplates.pebble.cache.PebbleCache;
14 16  import io.pebbletemplates.pebble.cache.tag.ConcurrentMapTagCache;
15 17  import io.pebbletemplates.pebble.cache.tag.NoOpTagCache;
16 18  import io.pebbletemplates.pebble.cache.template.ConcurrentMapTemplateCache;
17 19  import io.pebbletemplates.pebble.cache.template.NoOpTemplateCache;
18 20  import io.pebbletemplates.pebble.error.LoaderException;
   21 + import io.pebbletemplates.pebble.extension.*;
   22 + import io.pebbletemplates.pebble.extension.escaper.EscapingStrategy;
19 23  import io.pebbletemplates.pebble.lexer.LexerImpl;
20 24  import io.pebbletemplates.pebble.lexer.Syntax;
21 25  import io.pebbletemplates.pebble.lexer.TokenStream;
   26 + import io.pebbletemplates.pebble.loader.ClasspathLoader;
   27 + import io.pebbletemplates.pebble.loader.Loader;
   28 + import io.pebbletemplates.pebble.loader.StringLoader;
22 29  import io.pebbletemplates.pebble.node.RootNode;
23 30  import io.pebbletemplates.pebble.parser.Parser;
24 31  import io.pebbletemplates.pebble.parser.ParserImpl;
25 32  import io.pebbletemplates.pebble.parser.ParserOptions;
```

```
26        - import
            io.pebbletemplates.pebble.attributes.methodaccess.BlacklistMethodAccessValidato
            r;
27        - import io.pebbletemplates.pebble.attributes.methodaccess.MethodAccessValidator;
28        - import io.pebbletemplates.pebble.extension.escaper.EscapingStrategy;
29        - import io.pebbletemplates.pebble.loader.ClasspathLoader;
30        - import io.pebbletemplates.pebble.loader.DelegatingLoader;
31        - import io.pebbletemplates.pebble.loader.FileLoader;
32        - import io.pebbletemplates.pebble.loader.Loader;
33        - import io.pebbletemplates.pebble.loader.StringLoader;
34        - import io.pebbletemplates.pebble.extension.*;
35  33      import io.pebbletemplates.pebble.template.EvaluationOptions;
36  34      import io.pebbletemplates.pebble.template.PebbleTemplate;
37  35      import io.pebbletemplates.pebble.template.PebbleTemplateImpl;
      36  + import io.pebbletemplates.pebble.utils.TypeUtils;
      37  + import org.slf4j.Logger;
      38  + import org.slf4j.LoggerFactory;
38  39
39  40      import java.io.IOException;
40  41      import java.io.Reader;
41        - import java.util.ArrayList;
42        - import java.util.List;
43  42      import java.util.Locale;
44  43      import java.util.concurrent.ExecutorService;
45  44      import java.util.function.Function;
46  45
47        - import io.pebbletemplates.pebble.utils.TypeUtils;
48        - import org.slf4j.Logger;
49        - import org.slf4j.LoggerFactory;
50        -
51  46      /**
52  47       * The main class used for compiling templates. The PebbleEngine is responsible
            for delegating
53  48       * responsibility to the lexer, parser, compiler, and template cache.
          @@ -584,10 +579,7 @@ public PebbleEngine build() {
584 579
585 580           // default loader
586 581           if (this.loader == null) {
587       -           List<Loader<?>> defaultLoadingStrategies = new ArrayList<>();
```

```
588        -          defaultLoadingStrategies.add(new ClasspathLoader());
589        -          defaultLoadingStrategies.add(new FileLoader());
590        -          this.loader = new DelegatingLoader(defaultLoadingStrategies);
       582 +          this.loader = new ClasspathLoader();
591    583           }
592    584
593    585           // default locale
```

...bbletemplates/pebble/loader/FileLoader.java

```
@@ -10,18 +10,12 @@
10    10
11    11    import io.pebbletemplates.pebble.error.LoaderException;
12    12    import io.pebbletemplates.pebble.utils.PathUtils;
13       -
14    13    import org.slf4j.Logger;
15    14    import org.slf4j.LoggerFactory;
16    15
17       - import java.io.BufferedReader;
18       - import java.io.File;
19       - import java.io.FileInputStream;
20       - import java.io.FileNotFoundException;
21       - import java.io.InputStream;
22       - import java.io.InputStreamReader;
23       - import java.io.Reader;
24       - import java.io.UnsupportedEncodingException;
       16 + import java.io.*;
       17 + import java.nio.file.Path;
       18 + import java.nio.file.Paths;
25    19
26    20    /**
27    21     * This loader searches for a file located anywhere on the filesystem. It uses
              java.io.File to
          @@ -34,69 +28,35 @@ public class FileLoader implements Loader<String> {
34    28      private static final Logger logger =
              LoggerFactory.getLogger(FileLoader.class);
35    29
36    30      private String prefix;
37       -
38    31      private String suffix;
```

```
39        -
40  32      private String charset = "UTF-8";
41  33
        34  +   public FileLoader(String prefix) {
        35  +     this.setPrefix(prefix);
        36  +   }
        37  +
42  38      @Override
43  39      public Reader getReader(String templateName) {
44        -     // try to load File
45        -     InputStream is = null;
46  40        File file = this.getFile(templateName);
47        -     if (file.exists() && file.isFile()) {
48        -       try {
49        -         is = new FileInputStream(file);
50        -       } catch (FileNotFoundException e) {
51        -       }
52        -     }
53        -
54        -     if (is == null) {
55        -       throw new LoaderException(null,
56        -           "Could not find template \"" + templateName + "\"");
57        -     }
58        -
59  41        try {
        42  +       InputStream is = new FileInputStream(file);
60  43          return new BufferedReader(new InputStreamReader(is, this.charset));
        44  +     } catch (FileNotFoundException e) {
        45  +       throw new LoaderException(e, String.format("Could not find template
            [prefix='%s', templateName='%s']", this.prefix, templateName));
61  46      } catch (UnsupportedEncodingException e) {
        47  +       throw new LoaderException(e, String.format("Invalid charset '%s'",
            this.charset));
62  48        }
63        -
64        -     return null;
65  49      }
66  50
67  51      private File getFile(String templateName) {
68        -     // add the prefix and ensure the prefix ends with a separator character
```

```
69        -     StringBuilder path = new StringBuilder();
70        -     if (this.getPrefix() != null) {
71        -
72        -       path.append(this.getPrefix());
73        -
74        -       if (!this.getPrefix().endsWith(String.valueOf(File.separatorChar))) {
75        -         path.append(File.separatorChar);
76        -       }
77        -     }
78        -
79  52          templateName = templateName + (this.getSuffix() == null ? "" :
              this.getSuffix());
        53    +     templateName = PathUtils.sanitize(templateName, File.separatorChar);
80  54
81        -     logger.trace("Looking for template in {}{}.", path.toString(),
              templateName);
        55    +     Path path = Paths.get(this.getPrefix(), templateName);
        56    +     logger.trace("Looking for template in {}.", path);
82  57
83        -     /*
84        -      * if template name contains path segments, move those segments into the
85        -      * path variable. The below technique needs to know the difference
86        -      * between the path and file name.
87        -      */
88        -     String[] pathSegments = PathUtils.PATH_SEPARATOR_REGEX.split(templateName);
89        -
90        -     if (pathSegments.length > 1) {
91        -       // file name is the last segment
92        -       templateName = pathSegments[pathSegments.length - 1];
93        -     }
94        -     for (int i = 0; i < (pathSegments.length - 1); i++) {
95        -       path.append(pathSegments[i]).append(File.separatorChar);
96        -     }
97        -
98        -     // try to load File
99        -     return new File(path.toString(), templateName);
        58    +     this.checkIfDirectoryTraversal(templateName);
        59    +     return path.toFile();
100 60      }
101 61
```

```
102  62      public String getSuffix() {
          @@ -114,7 +74,17 @@ public String getPrefix() {
114  74
115  75      @Override
116  76      public void setPrefix(String prefix) {
117      -     this.prefix = prefix;
          +   if (prefix == null) {
          +     throw new LoaderException(null, "Prefix cannot be null");
          +   }
          +   String trimmedPrefix = prefix.trim();
          +   if (trimmedPrefix.isEmpty()) {
          +     throw new LoaderException(null, "Prefix cannot be empty");
          +   }
          +   if (!Paths.get(trimmedPrefix).isAbsolute()) {
          +     throw new LoaderException(null, "Prefix must be an absolute path");
          +   }
          +   this.prefix = trimmedPrefix;
118  88      }
119  89
120  90      public String getCharset() {
          @@ -140,4 +110,23 @@ public String createCacheKey(String templateName) {
140  110     public boolean resourceExists(String templateName) {
141  111       return this.getFile(templateName).exists();
142  112     }
          +
          +   private void checkIfDirectoryTraversal(String templateName) {
          +     Path baseDirPath = Paths.get(prefix);
          +     Path userPath = Paths.get(templateName);
          +     if (userPath.isAbsolute()) {
          +       throw new LoaderException(null, String.format("templateName '%s' must be
            relative", templateName));
          +     }
          +
          +     // Join the two paths together, then normalize so that any ".." elements
          +     // in the userPath can remove parts of baseDirPath.
          +     // (e.g. "/foo/bar/baz" + "../attack" -> "/foo/bar/attack")
          +     Path resolvedPath = baseDirPath.resolve(userPath).normalize();
          +
          +     // Make sure the resulting path is still within the required directory.
          +     // (In the example above, "/foo/bar/attack" is not.)
```

```
128 +     if (!resolvedPath.startsWith(baseDirPath)) {
129 +       throw new LoaderException(null, String.format("template is not in the
      base directory path [baseDir='%s', templateName='%s']", this.prefix,
      templateName));
130 +     }
131 +   }
143 132   }
```

...pebbletemplates/pebble/utils/PathUtils.java

```
@@ -44,7 +44,7 @@ public static String resolveRelativePath(String
relativePath, String anchorPath,
44 44       return null;
45 45     }
46 46
47    -   private static String sanitize(String path, char expectedSeparator) {
   47 +   public static String sanitize(String path, char expectedSeparator) {
48 48       return PATH_SEPARATOR_REGEX.matcher(path)
49 49
                .replaceAll(Matcher.quoteReplacement(String.valueOf(expectedSeparator)));
50 50     }
```

.../pebbletemplates/pebble/FileLoaderTest.java

```
...  @@ -0,0 +1,132 @@
1  + package io.pebbletemplates.pebble;
2  +
3  + import io.pebbletemplates.pebble.error.LoaderException;
4  + import io.pebbletemplates.pebble.error.PebbleException;
5  + import io.pebbletemplates.pebble.loader.FileLoader;
6  + import io.pebbletemplates.pebble.loader.Loader;
7  + import io.pebbletemplates.pebble.template.PebbleTemplate;
8  + import org.junit.jupiter.api.Test;
9  + import org.junit.jupiter.params.ParameterizedTest;
10 + import org.junit.jupiter.params.provider.ValueSource;
11 +
12 + import java.io.IOException;
13 + import java.io.StringWriter;
14 + import java.io.Writer;
15 + import java.net.URISyntaxException;
```

```
16 + import java.nio.file.Paths;
17 +
18 + import static org.junit.jupiter.api.Assertions.assertEquals;
19 + import static org.junit.jupiter.api.Assertions.assertThrows;
20 +
21 + public class FileLoaderTest {
22 +
23 +   @Test
24 +   void testFileLoaderPrefixNull() {
25 +     assertThrows(LoaderException.class, () -> new FileLoader(null));
26 +   }
27 +
28 +   @Test
29 +   void testFileLoaderPrefixEmpty() {
30 +     assertThrows(LoaderException.class, () -> new FileLoader(" "));
31 +   }
32 +
33 +   @Test
34 +   void testFileLoaderPrefixRelativePath() {
35 +     assertThrows(LoaderException.class, () -> new FileLoader(" ../bar "));
36 +   }
37 +
38 +   @Test
39 +   void testFileLoader() throws PebbleException, IOException, URISyntaxException
     {
40 +     String prefix =
     Paths.get(this.getClass().getClassLoader().getResource("templates").toURI()).to
     String();
41 +     Loader<?> loader = new FileLoader(prefix);
42 +     loader.setSuffix(".suffix");
43 +     PebbleEngine engine = new
     PebbleEngine.Builder().loader(loader).strictVariables(false).build();
44 +     PebbleTemplate template1 = engine.getTemplate("template.loaderTest.peb");
45 +     Writer writer1 = new StringWriter();
46 +     template1.evaluate(writer1);
47 +     assertEquals("SUCCESS", writer1.toString());
48 +   }
49 +
50 +   @Test
```

```
51 +   void testFileLoaderAbsoluteTemplateName() throws PebbleException,
   URISyntaxException {
52 +     String prefix =
   Paths.get(this.getClass().getClassLoader().getResource("templates").toURI()).to
   String();
53 +     Loader<?> loader = new FileLoader(prefix);
54 +     loader.setSuffix(".suffix");
55 +     PebbleEngine engine = new
   PebbleEngine.Builder().loader(loader).strictVariables(false).build();
56 +     assertThrows(LoaderException.class, () ->
   engine.getTemplate("/template.loaderTest.peb"));
57 +   }
58 +
59 +   @Test
60 +   void testFileLoaderTemplateNameIsADirectory() throws PebbleException,
   URISyntaxException {
61 +     String prefix =
   Paths.get(this.getClass().getClassLoader().getResource("templates").toURI()).to
   String();
62 +     Loader<?> loader = new FileLoader(prefix);
63 +     PebbleEngine engine = new
   PebbleEngine.Builder().loader(loader).strictVariables(false).build();
64 +     assertThrows(LoaderException.class, () -> engine.getTemplate("loader"));
65 +   }
66 +
67 +   @Test
68 +   void testFileLoaderRelativeTemplateName() throws PebbleException,
   IOException, URISyntaxException {
69 +     String prefix =
   Paths.get(this.getClass().getClassLoader().getResource("templates").toURI()).ge
   tParent().toString();
70 +     Loader<?> loader = new FileLoader(prefix);
71 +     loader.setSuffix(".suffix");
72 +     PebbleEngine engine = new
   PebbleEngine.Builder().loader(loader).strictVariables(false).build();
73 +     PebbleTemplate template1 =
   engine.getTemplate("templates/template.loaderTest.peb");
74 +     Writer writer1 = new StringWriter();
75 +     template1.evaluate(writer1);
76 +     assertEquals("SUCCESS", writer1.toString());
```

```
77  +   }
78  +
79  +   @Test
80  +   void testFileLoaderPathTraversal() throws PebbleException, URISyntaxException
      {
81  +     String prefix =
      Paths.get(this.getClass().getClassLoader().getResource("templates").toURI()).to
      String();
82  +     Loader<?> loader = new FileLoader(prefix);
83  +     loader.setSuffix(".peb");
84  +     PebbleEngine engine = new
      PebbleEngine.Builder().loader(loader).strictVariables(false).build();
85  +     assertThrows(LoaderException.class, () -> engine.getTemplate("../template-
      tests/DoubleNestedIfStatement"));
86  +   }
87  +
88  +   @ParameterizedTest
89  +   @ValueSource(strings = {"%2e%2e%2f", "%2e%2e/", "..%2f", "%2e%2e%5c",
      "%2e%2e\\", "..%5c", "%252e%252e%255c", "..%255c"})
90  +   void testFileLoaderPathTraversalEncoded(String relativePath) throws
      URISyntaxException {
91  +     String prefix =
      Paths.get(this.getClass().getClassLoader().getResource("templates").toURI()).to
      String();
92  +     Loader<?> loader = new FileLoader(prefix);
93  +     loader.setSuffix(".peb");
94  +     PebbleEngine engine = new
      PebbleEngine.Builder().loader(loader).strictVariables(false).build();
95  +     assertThrows(LoaderException.class, () -> engine.getTemplate(relativePath +
      "template-tests/DoubleNestedIfStatement"));
96  +   }
97  +
98  +   @Test
99  +   void testFileLoaderUnsupportedCharset() throws PebbleException,
      URISyntaxException {
100 +     String prefix =
      Paths.get(this.getClass().getClassLoader().getResource("templates").toURI()).to
      String();
101 +     Loader<?> loader = new FileLoader(prefix);
102 +     loader.setCharset("foobar");
```

```
103 +     PebbleEngine engine = new
    PebbleEngine.Builder().loader(loader).strictVariables(false).build();
104 +     assertThrows(LoaderException.class, () ->
    engine.getTemplate("template.loaderTest.peb"));
105 +   }
106 +
107 +   /**
108 +    * Tests if relative includes work. Issue #162.
109 +    */
110 +   @Test
111 +   void testFileLoaderPathWithBackslash() throws IOException, URISyntaxException
    {
112 +     String prefix =
    Paths.get(this.getClass().getClassLoader().getResource("templates").toURI()).to
    String();
113 +     PebbleEngine pebble = new PebbleEngine.Builder().loader(new
    FileLoader(prefix)).build();
114 +     PebbleTemplate template =
    pebble.getTemplate("relativepath/subdirectory1/template.forwardslashes.peb".rep
    lace("/", "\\")); // ensure backslashes in all environments
115 +     Writer writer = new StringWriter();
116 +     template.evaluate(writer);
117 +     assertEquals("included", writer.toString());
118 +   }
119 +
120 +   /**
121 +    * Issue #162.
122 +    */
123 +   @Test
124 +   void testFileLoaderPathWithForwardSlash() throws IOException,
    URISyntaxException {
125 +     String prefix =
    Paths.get(this.getClass().getClassLoader().getResource("templates").toURI()).to
    String();
126 +     PebbleEngine pebble = new PebbleEngine.Builder().loader(new
    FileLoader(prefix)).build();
127 +     PebbleTemplate template =
    pebble.getTemplate("relativepath/subdirectory1/template.backwardslashes.peb");
128 +     Writer writer = new StringWriter();
129 +     template.evaluate(writer);
```

```
+     assertEquals("included", writer.toString());
+   }
+ }
```

...a/io/pebbletemplates/pebble/LoaderTest.java

```
@@ -15,7 +15,6 @@
import org.junit.jupiter.api.Test;

import java.io.*;
- import java.net.URISyntaxException;
import java.net.URL;
import java.net.URLClassLoader;
import java.util.ArrayList;
@@ -69,19 +68,6 @@ void testClassLoaderLoaderWithNestedTemplateInJar() throws
PebbleException, IOEx

  }

-   @Test
-   void testFileLoader() throws PebbleException, IOException, URISyntaxException
{
-     Loader<?> loader = new FileLoader();
-     loader.setSuffix(".suffix");
-     PebbleEngine engine = new
PebbleEngine.Builder().loader(loader).strictVariables(false).build();
-     URL url = this.getClass().getResource("/templates/template.loaderTest.peb");
-     PebbleTemplate template1 = engine.getTemplate(new
File(url.toURI()).getPath());
-     Writer writer1 = new StringWriter();
-     template1.evaluate(writer1);
-     assertEquals("SUCCESS", writer1.toString());
-
-   }
-
  @Test
  void testDelegatingLoader() throws PebbleException, IOException {
    List<Loader<?>> loaders = new ArrayList<>();
```

...ebbletemplates/pebble/TestRelativePath.java

```
@@ -1,17 +1,12 @@
package io.pebbletemplates.pebble;

import io.pebbletemplates.pebble.error.PebbleException;
- import io.pebbletemplates.pebble.loader.FileLoader;
import io.pebbletemplates.pebble.template.PebbleTemplate;
-
import org.junit.jupiter.api.Test;

- import java.io.File;
import java.io.IOException;
import java.io.StringWriter;
import java.io.Writer;
- import java.net.URISyntaxException;
- import java.net.URL;

import static org.junit.jupiter.api.Assertions.assertEquals;

@@ -65,38 +60,4 @@ void testRelativeImports() throws PebbleException,
IOException {
        assertEquals("<input name=\"company\" value=\"forcorp\" type=\"text\" />",
            writer.toString().replaceAll("\\r?\\n", "").replace("\t", ""));
    }
-
-   /**
-    * Tests if relative includes work. Issue #162.
-    */
-   @Test
-   void testPathWithBackslashesWithRelativePathWithForwardSlashes()
-       throws PebbleException, IOException, URISyntaxException {
-     PebbleEngine pebble = new PebbleEngine.Builder().loader(new
FileLoader()).build();
-     URL url = this.getClass()
-
.getResource("/templates/relativepath/subdirectory1/template.forwardslashes.peb
");
-     PebbleTemplate template = pebble
-         .getTemplate(new File(url.toURI()).getPath()
```

```
80  -             .replace("/", "\\")); // ensure backslashes in all environments
81  -     Writer writer = new StringWriter();
82  -     template.evaluate(writer);
83  -     assertEquals("included", writer.toString());
84  -   }
85  -
86  -   /**
87  -    * Issue #162.
88  -    */
89  -   @Test
90  -   void testPathWithForwardSlashesWithRelativePathWithBackwardSlashes()
91  -       throws PebbleException, IOException, URISyntaxException {
92  -     PebbleEngine pebble = new PebbleEngine.Builder().loader(new
      FileLoader()).build();
93  -     URL url = this.getClass()
94  -
      .getResource("/templates/relativepath/subdirectory1/template.backwardslashes.pe
      b");
95  -     PebbleTemplate template = pebble
96  -         .getTemplate(new File(url.toURI()).getPath()
97  -             .replace("\\", "/")); // ensure forward slashes in all environments
98  -     Writer writer = new StringWriter();
99  -     template.evaluate(writer);
100 -     assertEquals("included", writer.toString());
101 -   }
102 63    }
```

pom.xml

```
    @@ -3,7 +3,7 @@
3   3     <modelVersion>4.0.0</modelVersion>
4   4     <groupId>io.pebbletemplates</groupId>
5   5     <artifactId>pebble-project</artifactId>
6       - <version>4.0.1-SNAPSHOT</version>
        6 + <version>4.1.0-SNAPSHOT</version>
7   7
8   8     <packaging>pom</packaging>
9   9
```

Comments 0

Please sign in to comment.