Item 348
git_comments:

git_commits:

1. **summary:** #670 fix PDF itests in native mode
   message: #670 fix PDF itests in native mode

github_issues:

1. **title:** PDF itest fails in native mode when building the font cache
   body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0) Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.get(HashMap.java:557) at com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```
2. **title:** PDF itest fails in native mode when building the font cache
   body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.

<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at
org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at
org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at
org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at
org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at
org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at
org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at
org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at
org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at
org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at
org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at
org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at
org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at
org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at
org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at
org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at
org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at
org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at
com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at
com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at
com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0)
Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image
runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object
java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from
entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at
org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at
org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at
com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at
java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at
javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at
javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at
java.util.HashMap.get(HashMap.java:557) at
com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at
com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at
com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at
com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at
com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at
com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```

3. **title:** PDF itest fails in native mode when building the font cache
body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0) Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at

com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.get(HashMap.java:557) at com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```

4. **title:** PDF itest fails in native mode when building the font cache
   body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0) Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.get(HashMap.java:557) at com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```

5. **title:** PDF itest fails in native mode when building the font cache
   body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at

org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0) Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.get(HashMap.java:557) at com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```

6. **title:** PDF itest fails in native mode when building the font cache
   body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0) Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at

org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.get(HashMap.java:557) at com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```

7. **title:** PDF itest fails in native mode when building the font cache
   body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0) Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.get(HashMap.java:557) at com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```

8. **title:** PDF itest fails in native mode when building the font cache
   body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at

org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0) Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.get(HashMap.java:557) at com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```
label: test

9. **title:** PDF itest fails in native mode when building the font cache
body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0) Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from

entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at
org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at
org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at
com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at
java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at
javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at
javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at
java.util.HashMap.get(HashMap.java:557) at
com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at
com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at
com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at
com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at
com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at
com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```

10. **title:** PDF itest fails in native mode when building the font cache
body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0) Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at java.util.HashMap.get(HashMap.java:557) at com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```
11. **title:** PDF itest fails in native mode when building the font cache
body: Having worked around [CAMEL-14452](https://issues.apache.org/jira/browse/CAMEL-14452), I now see the following when the PDF test is run in native mode with Camel 3.1.0. I think PDFBox tries to do some local font caching, which may be the cause of this. ``` Error: Detected a FileDescriptor in the image heap. File descriptors opened during image generation are no longer open at image run time, and the files might not even be present anymore at image run time. To see how this object got instantiated use -H:+TraceClassInitialization. The object was probably created by a class initializer and is reachable from a static field. You can request class initialization at image run time by using the option --initialize-at-build-time=<class-name>. Or you can write your own initialization methods and call them explicitly from your main entry point. Trace: object org.apache.fontbox.ttf.BufferedRandomAccessFile object org.apache.fontbox.ttf.RAFDataStream object org.apache.fontbox.ttf.TrueTypeFont object java.lang.ref.SoftReference object java.util.concurrent.ConcurrentHashMap$Node object java.util.concurrent.ConcurrentHashMap$Node[] object java.util.concurrent.ConcurrentHashMap object org.apache.pdfbox.pdmodel.font.FontCache object org.apache.pdfbox.pdmodel.font.FileSystemFontProvider method org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100() Call path from entry point to org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(): at

org.apache.pdfbox.pdmodel.font.FontMapperImpl$DefaultFontProvider.access$100(FontMapperImpl.java:126) at
org.apache.pdfbox.pdmodel.font.FontMapperImpl.getProvider(FontMapperImpl.java:147) at
org.apache.pdfbox.pdmodel.font.FontMapperImpl.findFont(FontMapperImpl.java:411) at
org.apache.pdfbox.pdmodel.font.FontMapperImpl.getTrueTypeFont(FontMapperImpl.java:319) at org.apache.pdfbox.pdmodel.font.PDTrueTypeFont.
<init>(PDTrueTypeFont.java:219) at org.apache.pdfbox.pdmodel.font.PDFontFactory.createFont(PDFontFactory.java:89) at
org.apache.pdfbox.pdmodel.PDResources.getFont(PDResources.java:146) at
org.apache.pdfbox.contentstream.operator.text.SetFontAndSize.process(SetFontAndSize.java:66) at
org.apache.pdfbox.contentstream.PDFStreamEngine.processOperator(PDFStreamEngine.java:875) at
org.apache.pdfbox.contentstream.PDFStreamEngine.processStreamOperators(PDFStreamEngine.java:509) at
org.apache.pdfbox.contentstream.PDFStreamEngine.processStream(PDFStreamEngine.java:483) at
org.apache.pdfbox.contentstream.PDFStreamEngine.processPage(PDFStreamEngine.java:156) at
org.apache.pdfbox.text.LegacyPDFStreamEngine.processPage(LegacyPDFStreamEngine.java:139) at
org.apache.pdfbox.text.PDFTextStripper.processPage(PDFTextStripper.java:391) at
org.apache.pdfbox.text.PDFTextStripper.processPages(PDFTextStripper.java:319) at
org.apache.pdfbox.text.PDFTextStripper.writeText(PDFTextStripper.java:266) at
org.apache.pdfbox.text.PDFTextStripper.getText(PDFTextStripper.java:227) at
org.apache.camel.component.pdf.PdfProducer.doExtractText(PdfProducer.java:109) at
org.apache.camel.component.pdf.PdfProducer.process(PdfProducer.java:71) at
org.apache.camel.support.AsyncProcessorConverterHelper$ProcessorToAsyncProcessorBridge.process(AsyncProcessorConverterHelper.java:67) at
org.apache.camel.processor.Resequencer.processExchange(Resequencer.java:318) at
org.apache.camel.processor.Resequencer$BatchSender.sendExchanges(Resequencer.java:546) at
org.apache.camel.processor.Resequencer$BatchSender.run(Resequencer.java:471) at
com.oracle.svm.core.thread.JavaThreads.threadStartRoutine(JavaThreads.java:460) at
com.oracle.svm.core.posix.thread.PosixJavaThreads.pthreadStartRoutine(PosixJavaThreads.java:193) at
com.oracle.svm.core.code.IsolateEnterStub.PosixJavaThreads_pthreadStartRoutine_e1f4a8c0039f8337338252cd8734f63a79b5e3df(generated:0)
Error: No instances of org.apache.pdfbox.pdmodel.font.PDType1Font are allowed in the image heap as this class should be initialized at image
runtime. To see how this object got instantiated use -H:+TraceClassInitialization. Trace: object java.util.HashMap$Node object
java.util.HashMap$Node[] object java.util.HashMap method org.apache.camel.component.pdf.Standard14Fonts.getByName(String) Call path from
entry point to org.apache.camel.component.pdf.Standard14Fonts.getByName(String): at
org.apache.camel.component.pdf.Standard14Fonts.getByName(Standard14Fonts.java:58) at
org.apache.camel.component.pdf.PdfConfiguration.setFont(PdfConfiguration.java:171) at
com.oracle.svm.reflect.PdfConfiguration_setFont_a95db4db2b78c2f69d084e6eac2d7ac87bee3c00_1326.invoke(Unknown Source) at
java.lang.reflect.Method.invoke(Method.java:498) at javax.enterprise.util.AnnotationLiteral.invoke(AnnotationLiteral.java:288) at
javax.enterprise.util.AnnotationLiteral.getMemberValue(AnnotationLiteral.java:276) at
javax.enterprise.util.AnnotationLiteral.hashCode(AnnotationLiteral.java:246) at java.util.HashMap.hash(HashMap.java:339) at
java.util.HashMap.get(HashMap.java:557) at
com.oracle.svm.jni.access.JNIReflectionDictionary.getClassObjectByName(JNIReflectionDictionary.java:123) at
com.oracle.svm.jni.functions.JNIFunctions.FindClass(JNIFunctions.java:326) at
com.oracle.svm.core.code.IsolateEnterStub.JNIFunctions_FindClass_3ec1032c6cb9443725d1e68194130533bfb04076(generated:0) at
com.oracle.graal.pointsto.constraints.UnsupportedFeatures.report(UnsupportedFeatures.java:133) at
com.oracle.graal.pointsto.BigBang.finish(BigBang.java:565) at
com.oracle.svm.hosted.NativeImageGenerator.runPointsToAnalysis(NativeImageGenerator.java:688) ... 7 more ```

github_issues_comments:

1. I saw that in the past but unable to reproduce reliably so its probably related to the font set-up of the system that runs the build (i.e. never seen that happening on CI)
2. Happens for me with the master branch too. You're right though, on CI builds it seems ok (not sure how). Some related issues: https://issues.apache.org/jira/browse/PDFBOX-4548 https://issues.apache.org/jira/browse/TIKA-2862
3. If one could reproduce this issue, I would be curious to see whether this [workaround] (https://github.com/quarkusio/quarkus/blob/master/extensions/tika/deployment/src/main/java/io/quarkus/tika/deployment/TikaProcessor.java#L69..L73l) would help ?
4. From what I can see, the only reason it passes on the CI builds is because the native build is done in Docker and the `ubi-quarkus-native-image` has no fonts installed. Therefore, no font loading or caching is attempted. It always fails for me when building the native executable on the host, since I do (obviously) have fonts installed. Using `RuntimeInitializedClassBuildItem` for `PDType1Font` on its own will not work. There's a bunch of dependent stuff that would also need to be initialized at runtime. We could maybe investigate that. The real problem is that there is way too much stuff being done via static initializers (including in the PDF component. See class `Standard14Fonts`)
5. create a PR to fix this issue here https://github.com/apache/camel-quarkus/pull/707
6. I tried to reproduce this issue locally by deleting the font cache to no avail. I was able to rebuild the font cache in quarkus jvm mode, hosted native mode and docker native mode, so the condition to reproduce this issue might be finer than that. I've tried the fix from @ffang locally, and still no issue :+1:. @jamesnetherton Do you still experience the issue when running pdf integration-tests from master ? Side note, I see a null pointer exception when restoring the font=Courier option in PdfResource but I think it could be handled in another quarkus issue that may be solved when upgrading to 3.1.0 where [CAMEL-14452|https://issues.apache.org/jira/browse/CAMEL-14452] is fixed.
7. Yes, it seems to work now. Thanks @ffang. Would be good to reinstate the `font=Courier` option in `PdfResource` after we upgrade to 3.1.0 to verify we definitely have fixed the problem(s).
8. **body:** ok, I propose to close this issue and open another one to restore the test coverage for the font option in 3.1.0 then. Well done guys :)
   label: test
9. @jamesnetherton @aldettinger Thanks guys! And without font=Courier, the default font PDType1Font.HELVETICA is used, so I believe the pdf font initialisition in native mode should work. And yes, we should retry with font=Courier after upgrading to camel 3.1.0

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Unable to configure font or page size on PDF endpoints
   description: This could be related to CAMEL-14284. If you try to set the font or pageSize options on a PDF endpoint then you get an NPE. E.g pdf:create?fontSize=6&pageSize=PAGE_SIZE_A5&font=courier Probably due to PdfEndpointConfigurer expecting the type to be either PDFont or PDRectangle when its actually trying to deal with a String.

jira_issues_comments: