CROZ

Obrada “velikih” XML dokumenata

Filip Bebek

16.5.2022.

"Some languages can be read by human, but not by machines, while others can be read by machines but not by humans. XML solves this problem by being readable to neither."

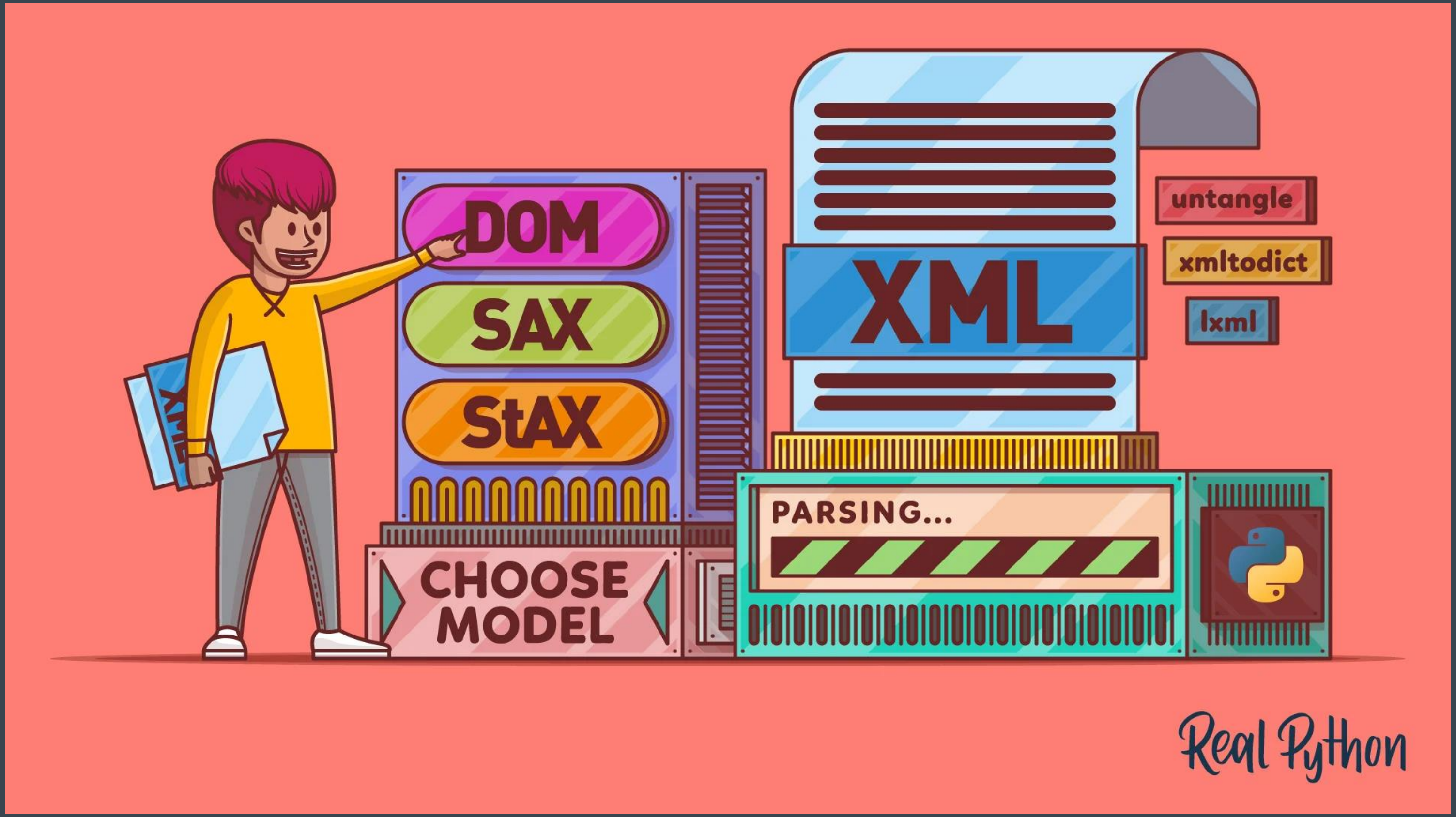
Poslovni zahtjev?

- Zaprimanje XML dokumenata veličine i preko 50 MB
- Izvlačenje podataka i njihova obrada
- Neki podaci unutar XML moguće veličine i do nekoliko MB
- Kreiranje novih XML dokumenata

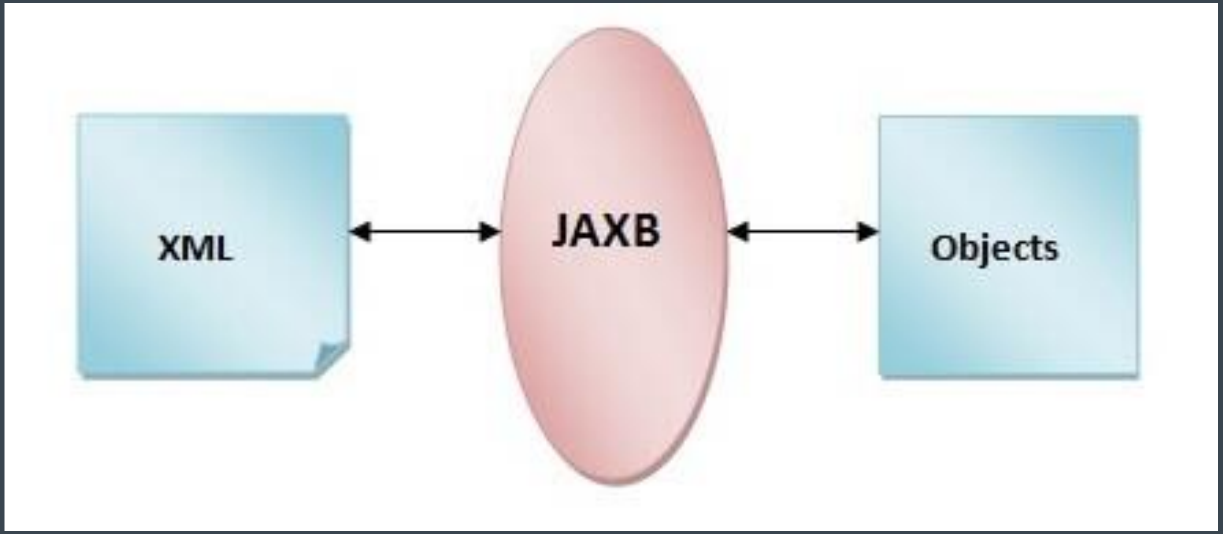
$$\begin{tikzcd}\boxed{A} \arrow[r, "x"] & \boxed{B} \arrow[r, "y"] & \boxed{C}\end{tikzcd}$$

Moguća rješenja

- JAXB
- SAX
- StAX



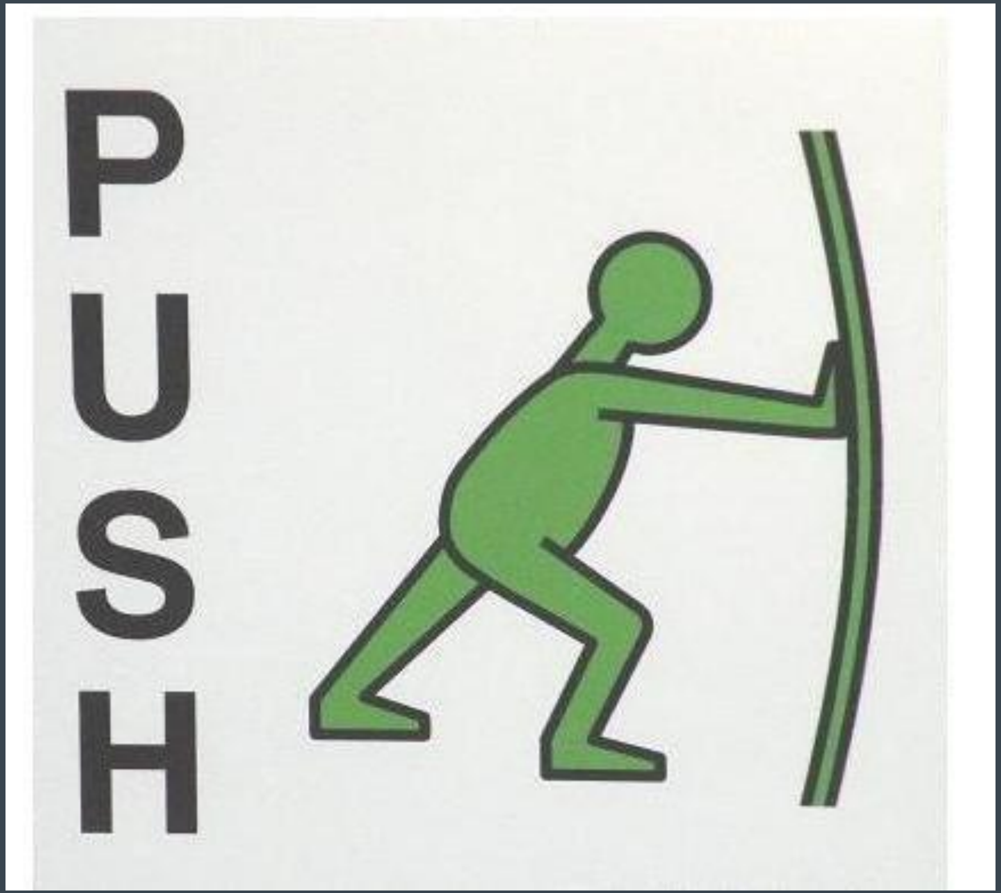
JAXB - Java Architecture for XML Binding

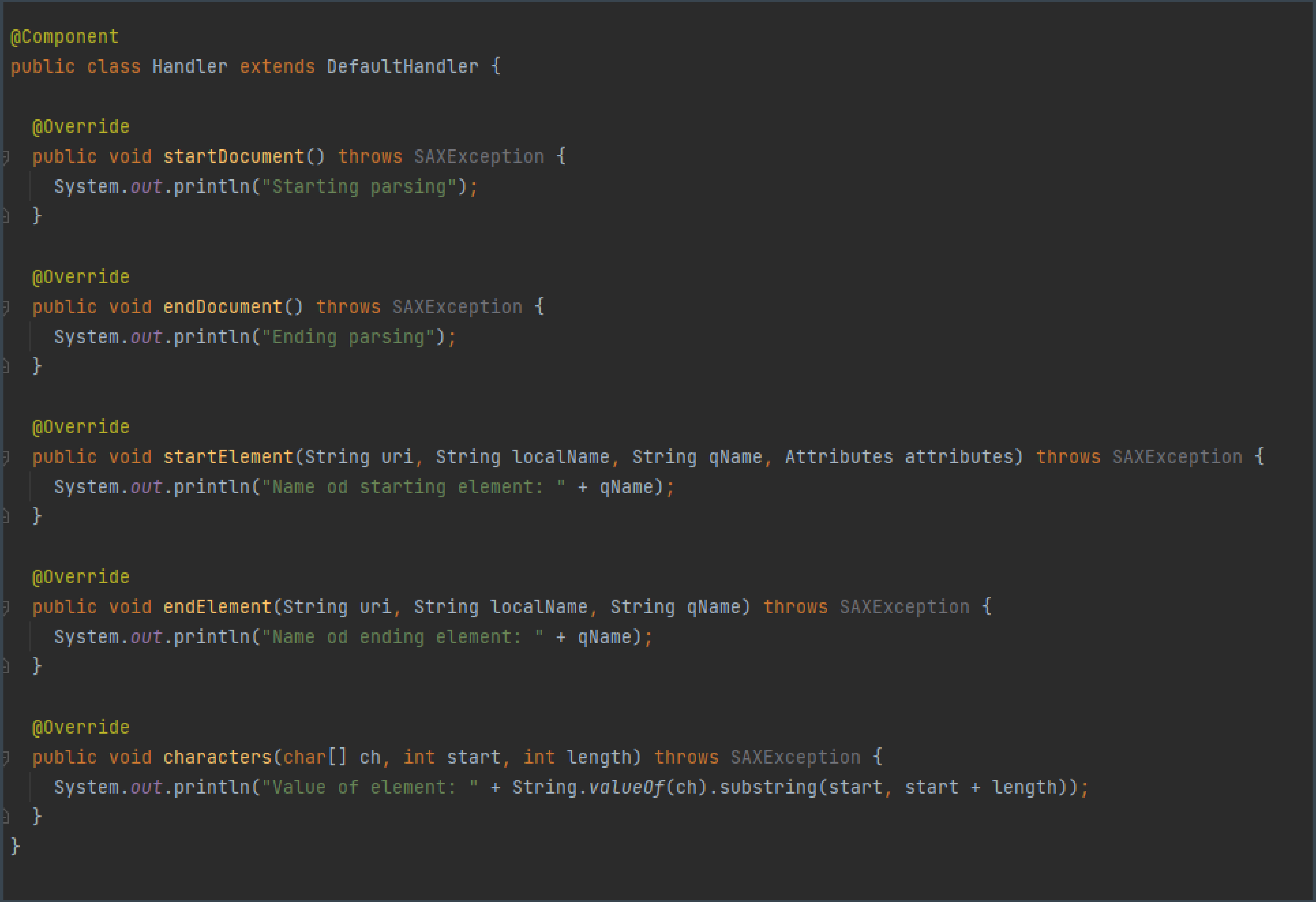


- Higher level API
- Mapira podatke iz XML dokumenata u Java objekte
- Standard za obradu malih XML dokumenata
- Učitavanje cijelog XML dokumenta u memoriju – PROBLEM!!

SAX – Simple API for XML

- Streamanje
- Baziran na eventima
- Push princip
- Memory efficiency
- Validacija shema




```java
@Component
public class Handler extends DefaultHandler {

  @Override
  public void startDocument() throws SAXException {
    System.out.println("Starting parsing");
  }

  @Override
  public void endDocument() throws SAXException {
    System.out.println("Ending parsing");
  }

  @Override
  public void startElement(String uri, String localName, String qName, Attributes attributes) throws SAXException {
    System.out.println("Name od starting element: " + qName);
  }

  @Override
  public void endElement(String uri, String localName, String qName) throws SAXException {
    System.out.println("Name od ending element: " + qName);
  }

  @Override
  public void characters(char[] ch, int start, int length) throws SAXException {
    System.out.println("Value of element: " + String.valueOf(ch).substring(start, start + length));
  }
}
```

Nedostatci

- Writing API
- Subparsing
- Jedna poruka po dretvi

StAX – Streaming API for XML

- Streamanje
- Iterator API && Cursor API
- Pull princip
- Memory effiency
- Nema validaciju shema

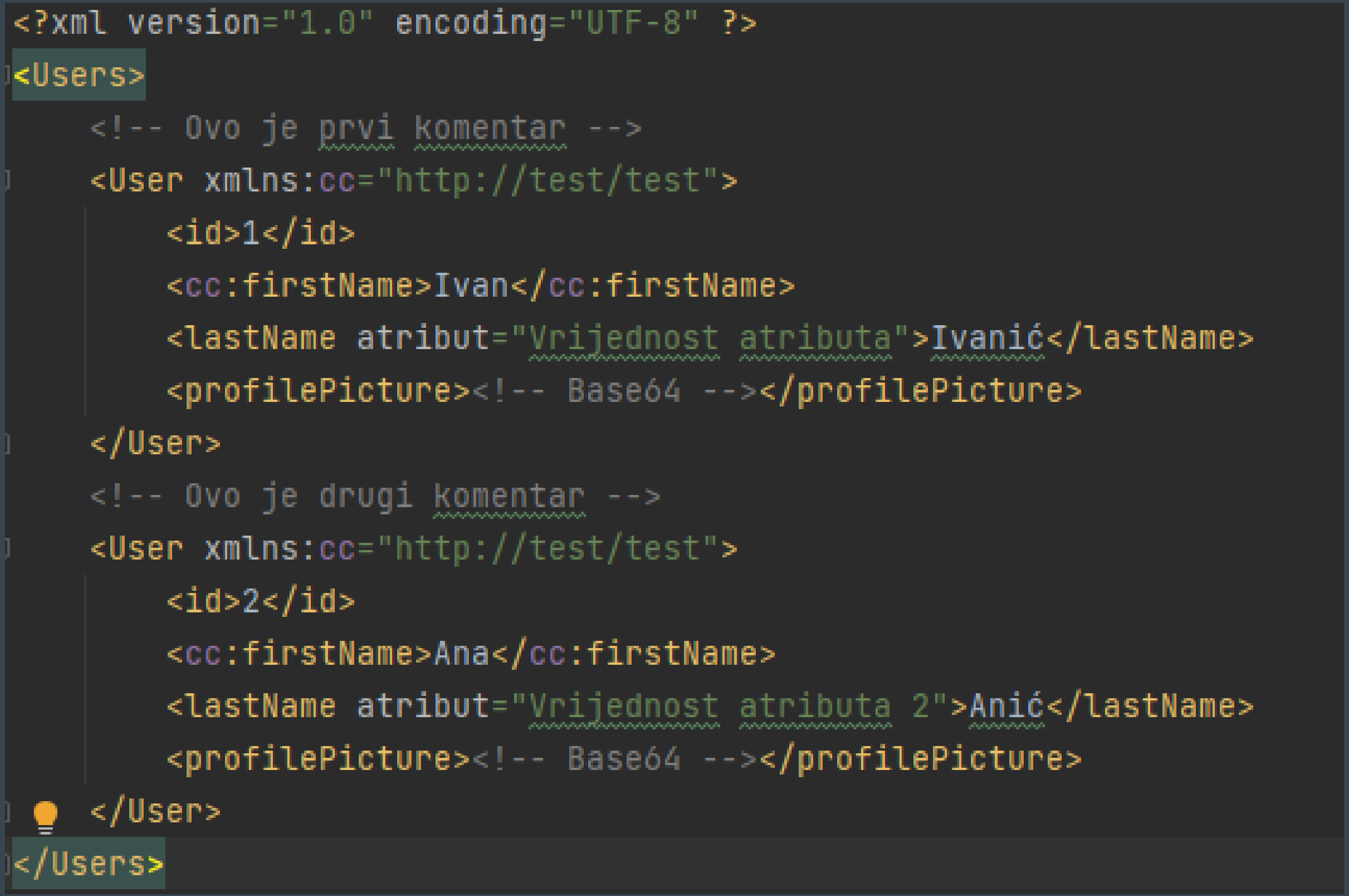
Cursor API

- Kursor za kretanje po XML-u
- XMLStreamReader
- XMLStreamWriter
- Efikasniji i brži kod

Iterator API

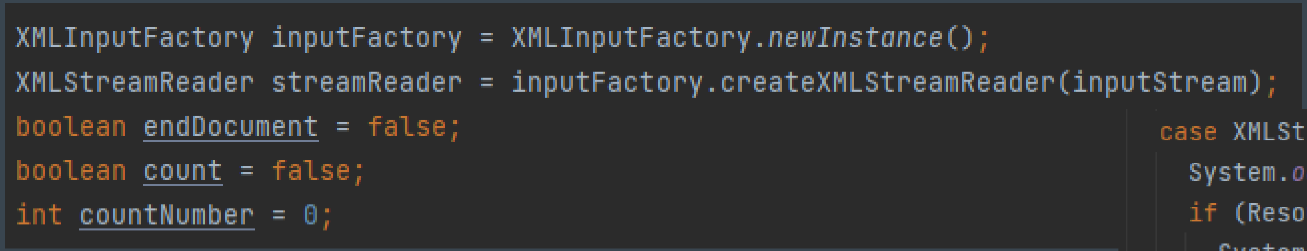
- Kao iterator kroz set evenata
- XMLEventReader
- XMLEventWriter
- Dodavanje/brisanje evenata

Reading API

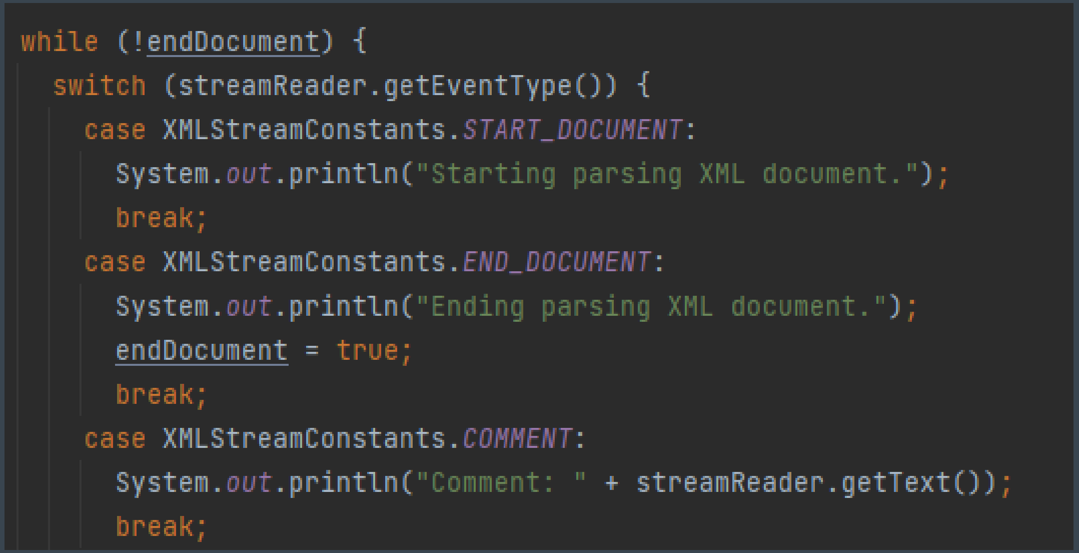


```xml
<?xml version="1.0" encoding="UTF-8" ?>
<Users>
    <!-- Ovo je prvi komentar -->
    <User xmlns:cc="http://test/test">
        <id>1</id>
        <cc:firstName>Ivan</cc:firstName>
        <lastName atribut="Vrijednost atributa">Ivanić</lastName>
        <profilePicture><!-- Base64 --></profilePicture>
    </User>
    <!-- Ovo je drugi komentar -->
    <User xmlns:cc="http://test/test">
        <id>2</id>
        <cc:firstName>Ana</cc:firstName>
        <lastName atribut="Vrijednost atributa 2">Anić</lastName>
        <profilePicture><!-- Base64 --></profilePicture>
    </User>
</Users>
```

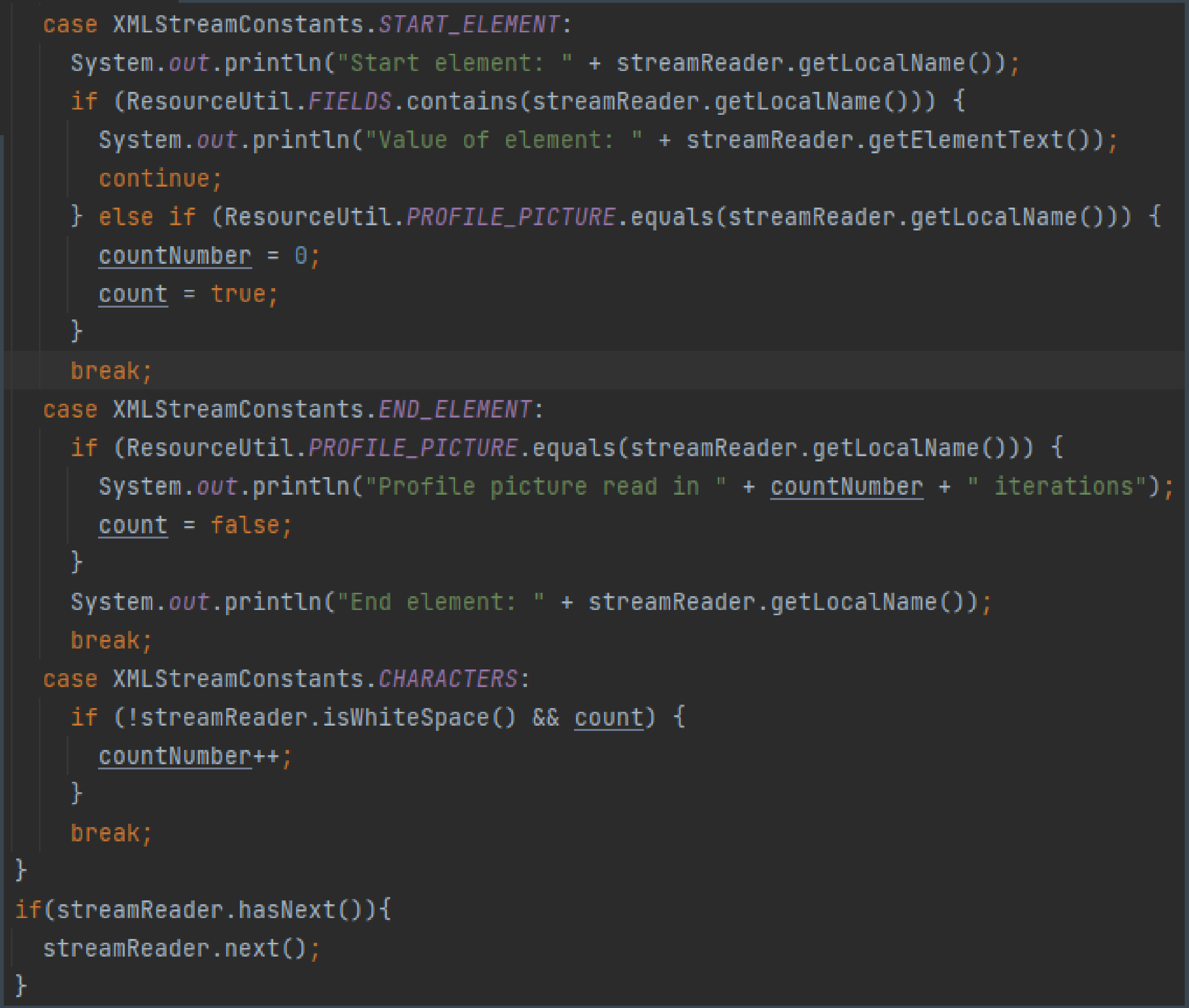
XMLStreamReader



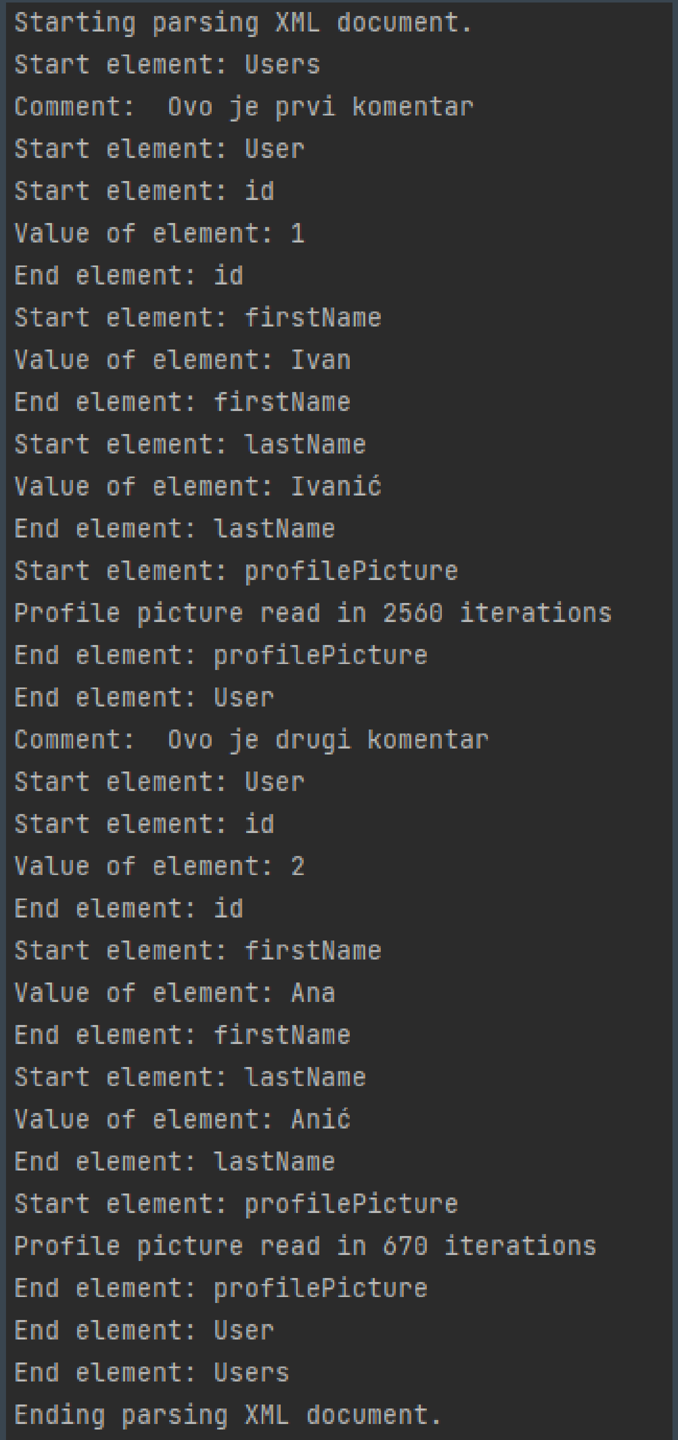
```java
XMLInputFactory inputFactory = XMLInputFactory.newInstance();
XMLStreamReader streamReader = inputFactory.createXMLStreamReader(inputStream);
boolean endDocument = false;
boolean count = false;
int countNumber = 0;
```



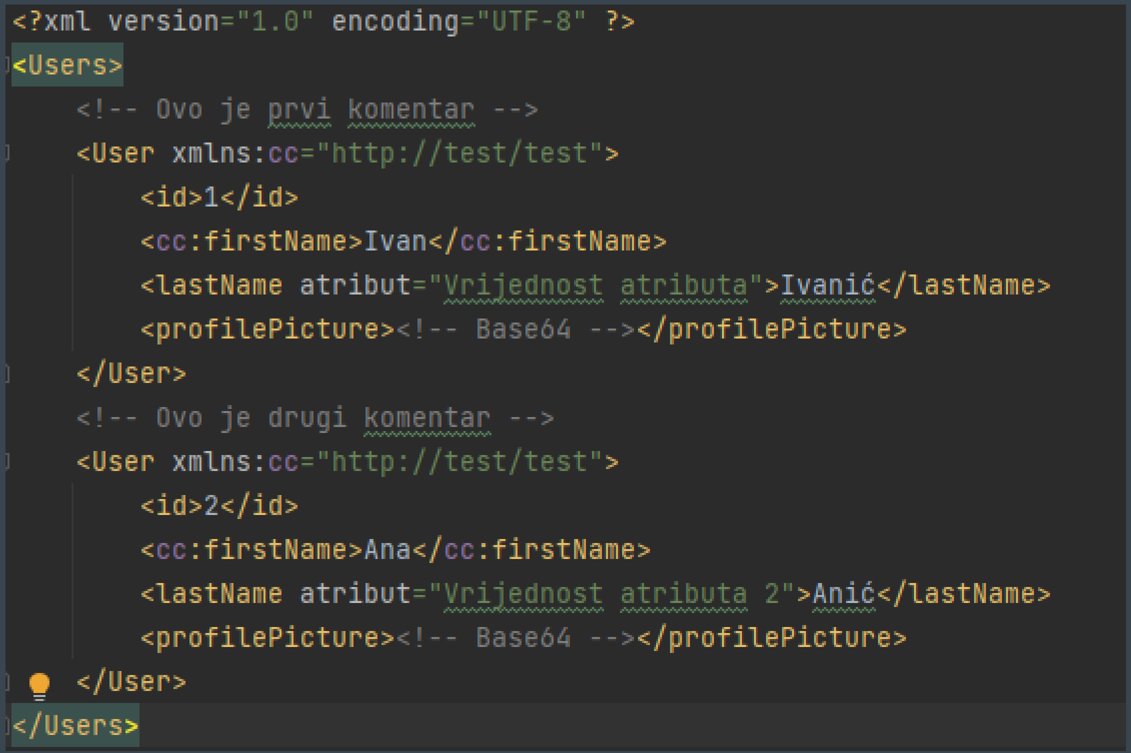
```java
while (!endDocument) {
  switch (streamReader.getEventType()) {
    case XMLStreamConstants.START_DOCUMENT:
      System.out.println("Starting parsing XML document.");
      break;
    case XMLStreamConstants.END_DOCUMENT:
      System.out.println("Ending parsing XML document.");
      endDocument = true;
      break;
    case XMLStreamConstants.COMMENT:
      System.out.println("Comment: " + streamReader.getText());
      break;
```



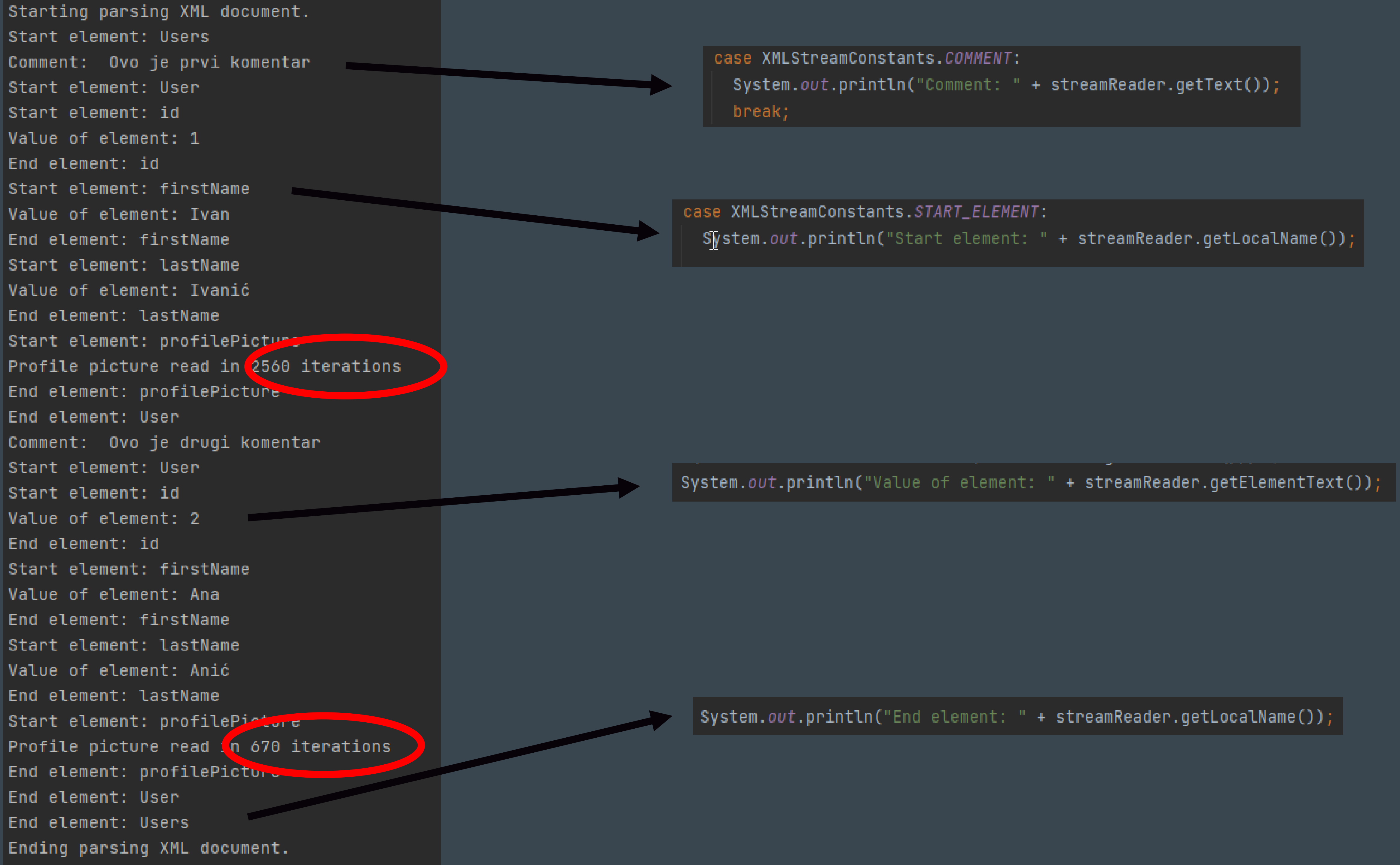
```java
    case XMLStreamConstants.START_ELEMENT:
      System.out.println("Start element: " + streamReader.getLocalName());
      if (ResourceUtil.FIELDS.contains(streamReader.getLocalName())) {
        System.out.println("Value of element: " + streamReader.getElementText());
        continue;
      } else if (ResourceUtil.PROFILE_PICTURE.equals(streamReader.getLocalName())) {
        countNumber = 0;
        count = true;
      }
      break;
    case XMLStreamConstants.END_ELEMENT:
      if (ResourceUtil.PROFILE_PICTURE.equals(streamReader.getLocalName())) {
        System.out.println("Profile picture read in " + countNumber + " iterations");
        count = false;
      }
      System.out.println("End element: " + streamReader.getLocalName());
      break;
    case XMLStreamConstants.CHARACTERS:
      if (!streamReader.isWhiteSpace() && count) {
        countNumber++;
      }
      break;
  }
  if(streamReader.hasNext()){
    streamReader.next();
  }
```



```
Starting parsing XML document.
Start element: Users
Comment:  Ovo je prvi komentar
Start element: User
Start element: id
Value of element: 1
End element: id
Start element: firstName
Value of element: Ivan
End element: firstName
Start element: lastName
Value of element: Ivanić
End element: lastName
Start element: profilePicture
Profile picture read in 2560 iterations
End element: profilePicture
End element: User
Comment:  Ovo je drugi komentar
Start element: User
Start element: id
Value of element: 2
End element: id
Start element: firstName
Value of element: Ana
End element: firstName
Start element: lastName
Value of element: Anić
End element: lastName
Start element: profilePicture
Profile picture read in 670 iterations
End element: profilePicture
End element: User
End element: Users
Ending parsing XML document.
```



```
<?xml version="1.0" encoding="UTF-8" ?>
<Users>
    <!-- Ovo je prvi komentar -->
    <User xmlns:cc="http://test/test">
        <id>1</id>
        <cc:firstName>Ivan</cc:firstName>
        <lastName atribut="Vrijednost atributa">Ivanić</lastName>
        <profilePicture><!-- Base64 --></profilePicture>
    </User>
    <!-- Ovo je drugi komentar -->
    <User xmlns:cc="http://test/test">
        <id>2</id>
        <cc:firstName>Ana</cc:firstName>
        <lastName atribut="Vrijednost atributa 2">Anić</lastName>
        <profilePicture><!-- Base64 --></profilePicture>
    </User>
</Users>
```



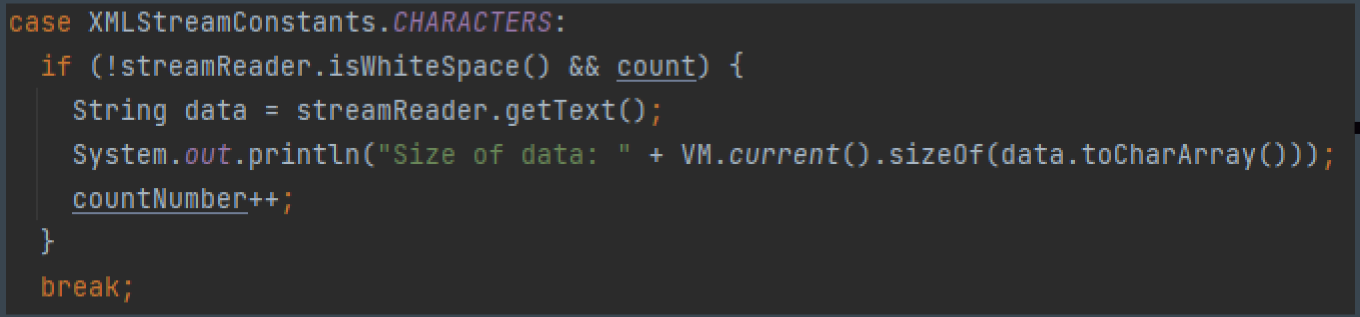
```
Starting parsing XML document.
Start element: Users
Comment:  Ovo je prvi komentar
Start element: User
Start element: id
Value of element: 1
End element: id
Start element: firstName
Value of element: Ivan
End element: firstName
Start element: lastName
Value of element: Ivanić
End element: lastName
Start element: profilePicture
Profile picture read in 2560 iterations
End element: profilePicture
End element: User
Comment:  Ovo je drugi komentar
Start element: User
Start element: id
Value of element: 2
End element: id
Start element: firstName
Value of element: Ana
End element: firstName
Start element: lastName
Value of element: Anić
End element: lastName
Start element: profilePicture
Profile picture read in 670 iterations
End element: profilePicture
End element: User
End element: Users
Ending parsing XML document.
```

```java
case XMLStreamConstants.COMMENT:
  System.out.println("Comment: " + streamReader.getText());
  break;
```

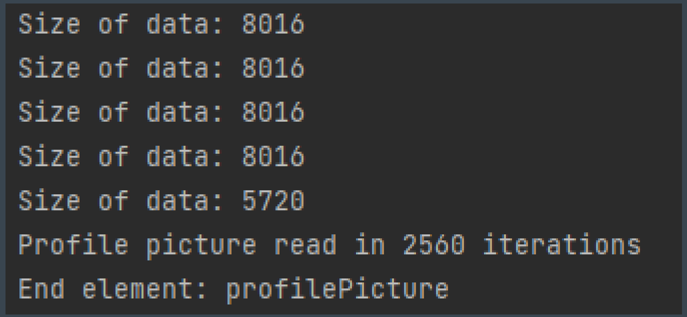
```java
case XMLStreamConstants.START_ELEMENT:
  System.out.println("Start element: " + streamReader.getLocalName());
```

```java
System.out.println("Value of element: " + streamReader.getElementText());
```

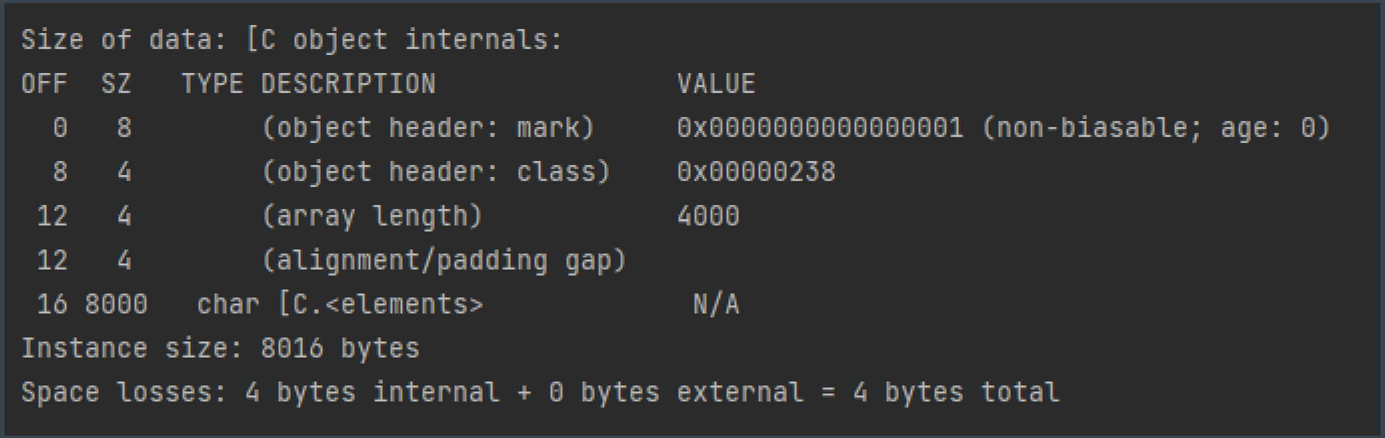
```java
System.out.println("End element: " + streamReader.getLocalName());
```



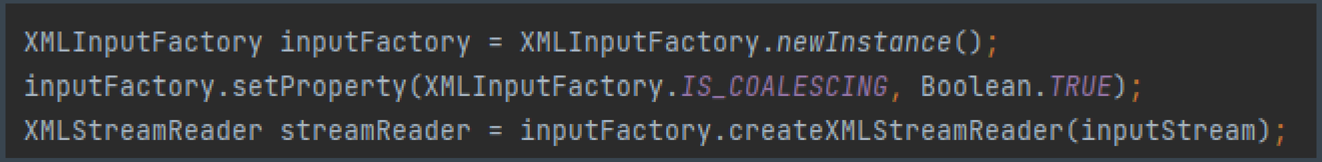
```java
case XMLStreamConstants.CHARACTERS:
  if (!streamReader.isWhiteSpace() && count) {
    String data = streamReader.getText();
    System.out.println("Size of data: " + VM.current().sizeOf(data.toCharArray()));
    countNumber++;
  }
  break;
```



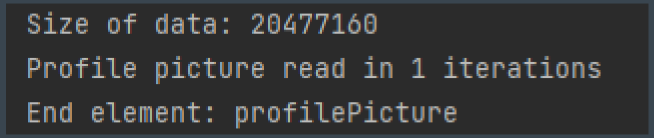
```
Size of data: 8016
Size of data: 8016
Size of data: 8016
Size of data: 8016
Size of data: 5720
Profile picture read in 2560 iterations
End element: profilePicture
```



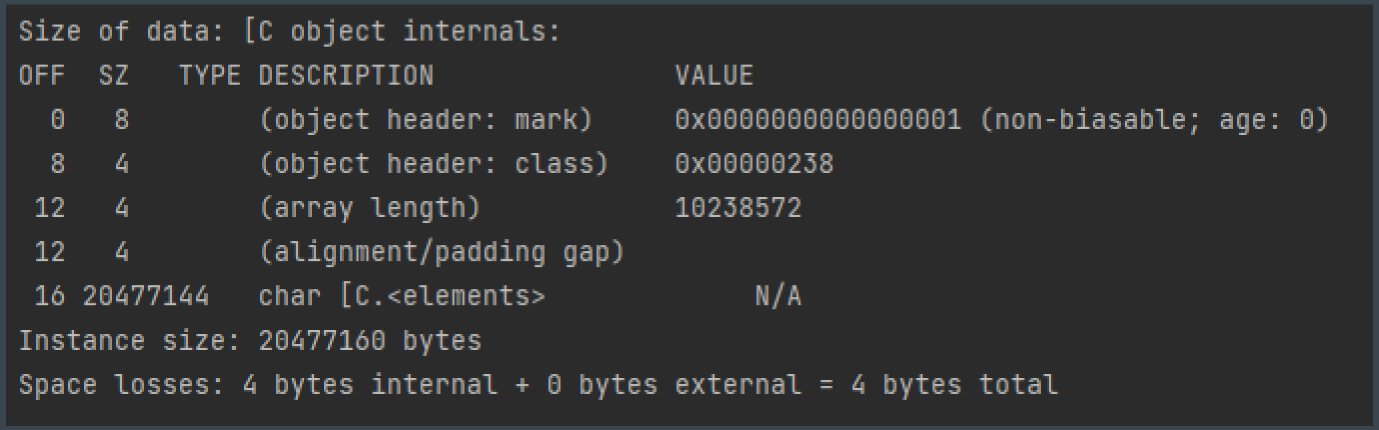
```
Size of data: [C object internals:
OFF  SZ   TYPE DESCRIPTION               VALUE
  0   8        (object header: mark)     0x0000000000000001 (non-biasable; age: 0)
  8   4        (object header: class)    0x00000238
 12   4        (array length)            4000
 12   4        (alignment/padding gap)
 16 8000   char [C.<elements>             N/A
Instance size: 8016 bytes
Space losses: 4 bytes internal + 0 bytes external = 4 bytes total
```



```java
XMLInputFactory inputFactory = XMLInputFactory.newInstance();
inputFactory.setProperty(XMLInputFactory.IS_COALESCING, Boolean.TRUE);
XMLStreamReader streamReader = inputFactory.createXMLStreamReader(inputStream);
```

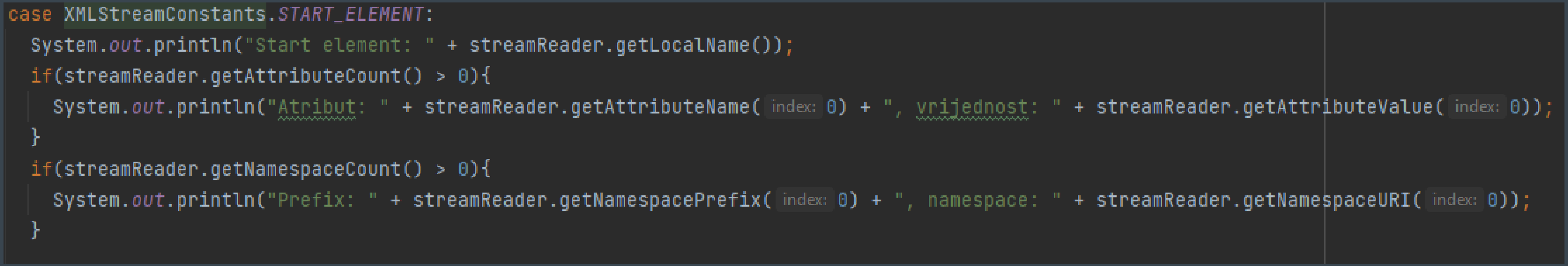


```
Size of data: 20477160
Profile picture read in 1 iterations
End element: profilePicture
```

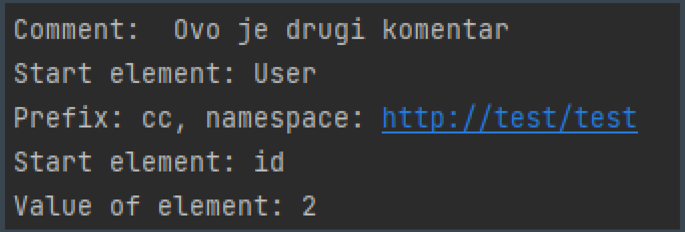


```
Size of data: [C object internals:
OFF  SZ   TYPE DESCRIPTION               VALUE
  0   8        (object header: mark)     0x0000000000000001 (non-biasable; age: 0)
  8   4        (object header: class)    0x00000238
 12   4        (array length)            10238572
 12   4        (alignment/padding gap)
 16 20477144   char [C.<elements>             N/A
Instance size: 20477160 bytes
Space losses: 4 bytes internal + 0 bytes external = 4 bytes total
```

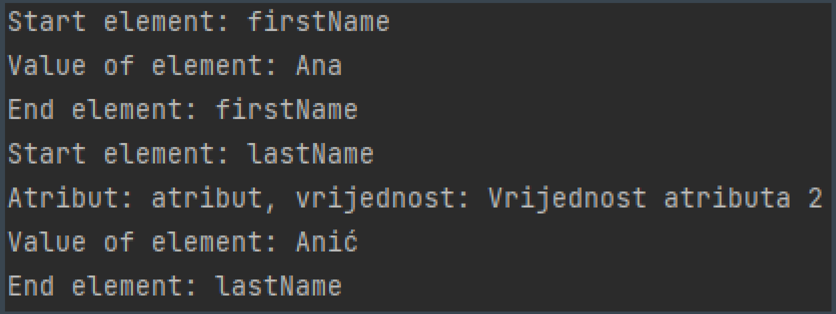
Atributi i Namespaceovi



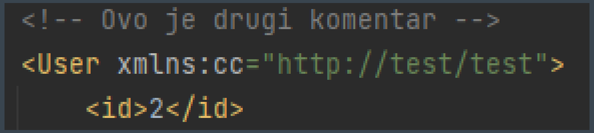
```java
case XMLStreamConstants.START_ELEMENT:
  System.out.println("Start element: " + streamReader.getLocalName());
  if(streamReader.getAttributeCount() > 0){
    System.out.println("Atribut: " + streamReader.getAttributeName( index: 0) + ", vrijednost: " + streamReader.getAttributeValue( index: 0));
  }
  if(streamReader.getNamespaceCount() > 0){
    System.out.println("Prefix: " + streamReader.getNamespacePrefix( index: 0) + ", namespace: " + streamReader.getNamespaceURI( index: 0));
  }
```



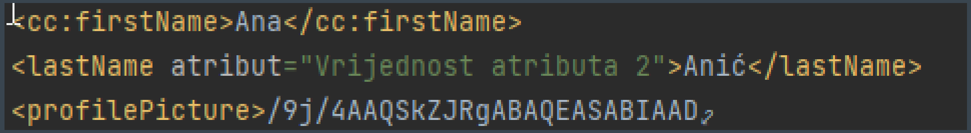
```
Comment:  Ovo je drugi komentar
Start element: User
Prefix: cc, namespace: http://test/test
Start element: id
Value of element: 2
```



```
Start element: firstName
Value of element: Ana
End element: firstName
Start element: lastName
Atribut: atribut, vrijednost: Vrijednost atributa 2
Value of element: Anić
End element: lastName
```



```xml
<!-- Ovo je drugi komentar -->
<User xmlns:cc="http://test/test">
    <id>2</id>
```



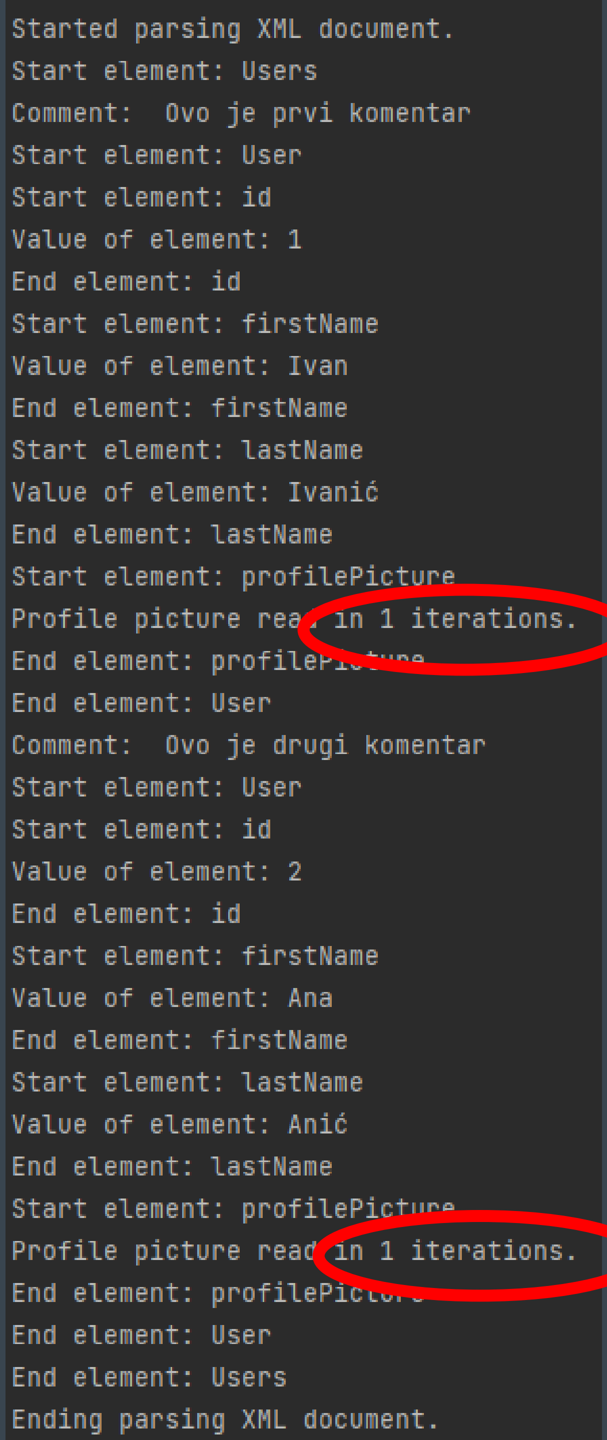
```xml
<cc:firstName>Ana</cc:firstName>
<lastName atribut="Vrijednost atributa 2">Anić</lastName>
<profilePicture>/9j/4AAQSkZJRgABAQEASABIAAD
```

XMLEventReader

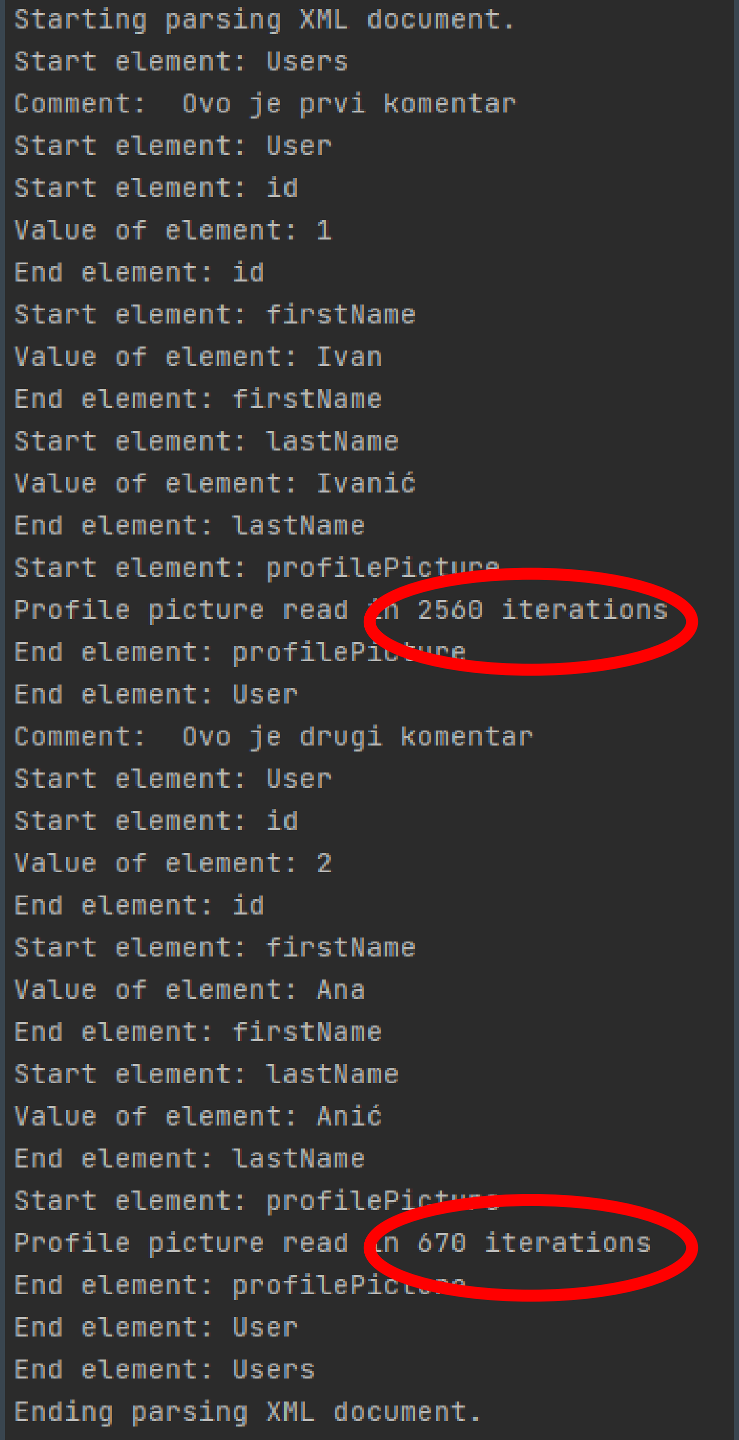
```java
XMLInputFactory inputFactory = XMLInputFactory.newInstance();
XMLEventReader eventReader = inputFactory.createXMLEventReader(inputStream);
boolean count = false;
int countNumber = 0;
```

```java
while (eventReader.hasNext()) {
  XMLEvent event = eventReader.nextEvent();
  switch (event.getEventType()) {
    case XMLStreamConstants.START_DOCUMENT:
      System.out.println("Started parsing XML document.");
      break;
    case XMLStreamConstants.END_DOCUMENT:
      System.out.println("Ending parsing XML document.");
      break;
    case XMLStreamConstants.COMMENT:
      Comment comment = (Comment) event;
      System.out.println("Comment: " + comment.getText());
      break;
```

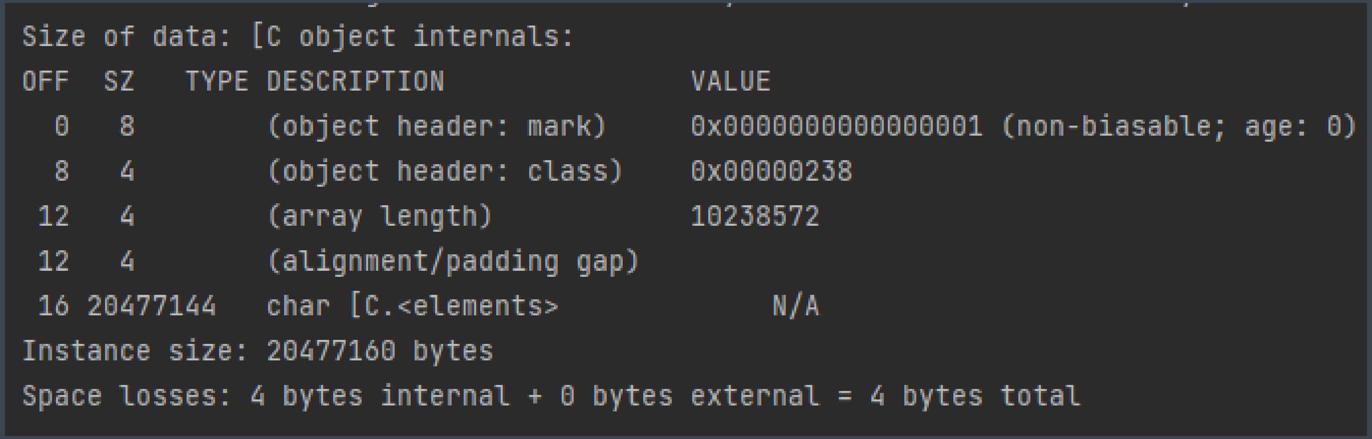
```java
case XMLStreamConstants.START_ELEMENT:
  StartElement startElement = event.asStartElement();
  System.out.println("Start element: " + startElement.getName().getLocalPart());
  if (ResourceUtil.FIELDS.contains(startElement.getName().getLocalPart())) {
    Characters characters = eventReader.nextEvent().asCharacters();
    System.out.println("Value of element: " + characters.getData());
  } else if (ResourceUtil.PROFILE_PICTURE.equals(startElement.getName().getLocalPart())) {
    countNumber = 0;
    count = true;
  }
  break;
case XMLStreamConstants.END_ELEMENT:
  EndElement endElement = event.asEndElement();
  if (ResourceUtil.PROFILE_PICTURE.equals(endElement.getName().getLocalPart())) {
    System.out.println("Profile picture read in " + countNumber + " iterations.");
    count = false;
  }
  System.out.println("End element: " + endElement.getName().getLocalPart());
  break;
case XMLStreamConstants.CHARACTERS:
  Characters characters = event.asCharacters();
  if (!characters.isWhiteSpace() && count) {
    countNumber++;
  }
  break;
}
```



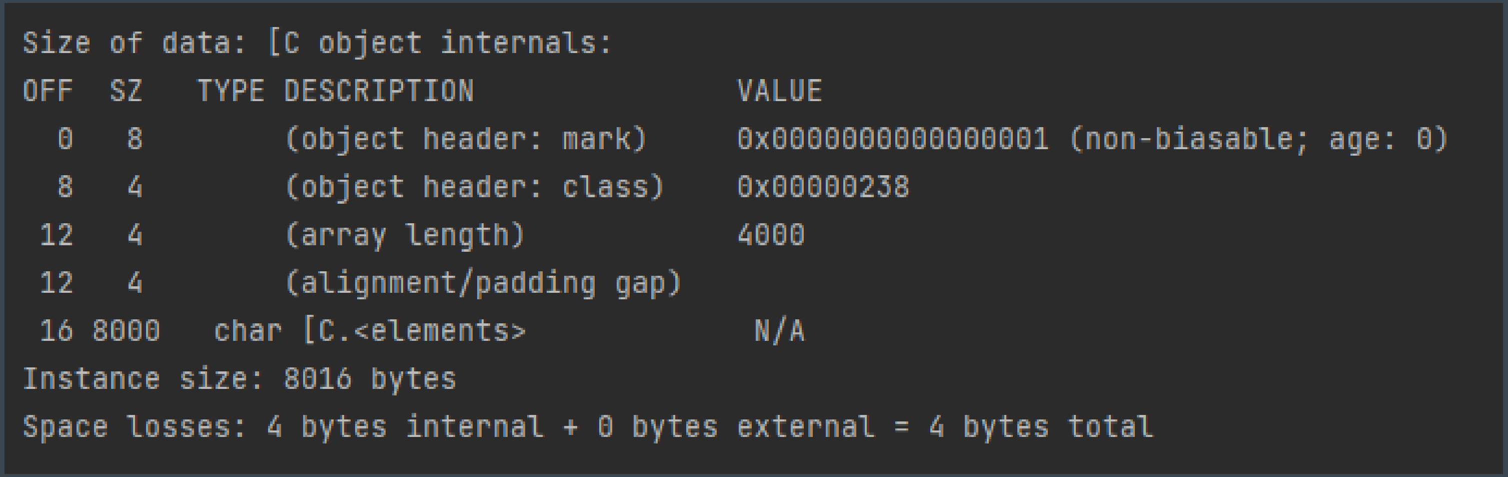
```
Started parsing XML document.
Start element: Users
Comment:  Ovo je prvi komentar
Start element: User
Start element: id
Value of element: 1
End element: id
Start element: firstName
Value of element: Ivan
End element: firstName
Start element: lastName
Value of element: Ivanić
End element: lastName
Start element: profilePicture
Profile picture read in 1 iterations.
End element: profilePicture
End element: User
Comment:  Ovo je drugi komentar
Start element: User
Start element: id
Value of element: 2
End element: id
Start element: firstName
Value of element: Ana
End element: firstName
Start element: lastName
Value of element: Anić
End element: lastName
Start element: profilePicture
Profile picture read in 1 iterations.
End element: profilePicture
End element: User
End element: Users
Ending parsing XML document.
```



```
Starting parsing XML document.
Start element: Users
Comment:  Ovo je prvi komentar
Start element: User
Start element: id
Value of element: 1
End element: id
Start element: firstName
Value of element: Ivan
End element: firstName
Start element: lastName
Value of element: Ivanić
End element: lastName
Start element: profilePicture
Profile picture read in 2560 iterations
End element: profilePicture
End element: User
Comment:  Ovo je drugi komentar
Start element: User
Start element: id
Value of element: 2
End element: id
Start element: firstName
Value of element: Ana
End element: firstName
Start element: lastName
Value of element: Anić
End element: lastName
Start element: profilePicture
Profile picture read in 670 iterations
End element: profilePicture
End element: User
End element: Users
Ending parsing XML document.
```

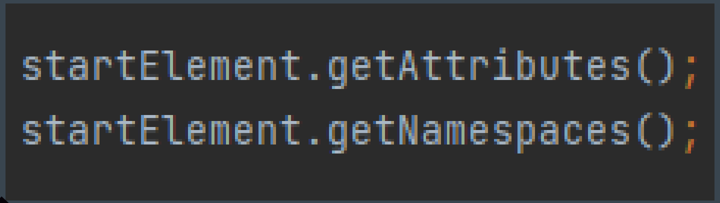


```
Size of data: [C object internals:
OFF  SZ   TYPE DESCRIPTION               VALUE
  0   8        (object header: mark)     0x0000000000000001 (non-biasable; age: 0)
  8   4        (object header: class)    0x00000238
 12   4        (array length)            10238572
 12   4        (alignment/padding gap)
 16 20477144   char [C.<elements>             N/A
Instance size: 20477160 bytes
Space losses: 4 bytes internal + 0 bytes external = 4 bytes total
```

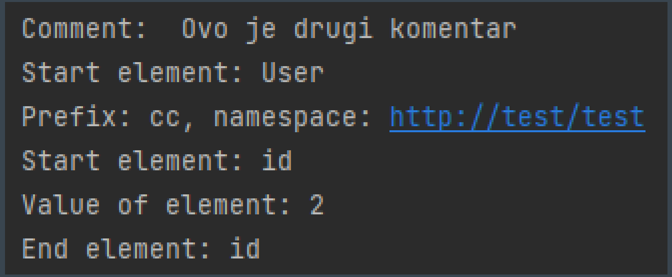


```
Size of data: [C object internals:
OFF  SZ   TYPE DESCRIPTION               VALUE
  0   8        (object header: mark)     0x0000000000000001 (non-biasable; age: 0)
  8   4        (object header: class)    0x00000238
 12   4        (array length)            4000
 12   4        (alignment/padding gap)
 16 8000   char [C.<elements>             N/A
Instance size: 8016 bytes
Space losses: 4 bytes internal + 0 bytes external = 4 bytes total
```

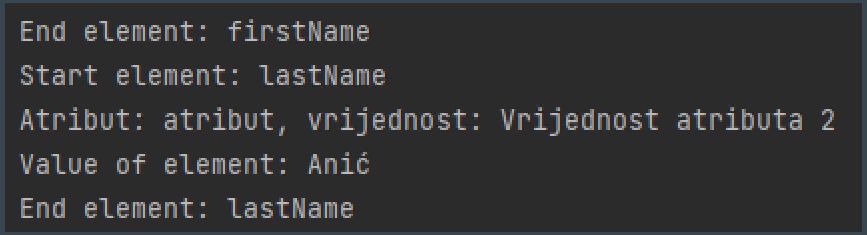
Atributi i Namespaceovi



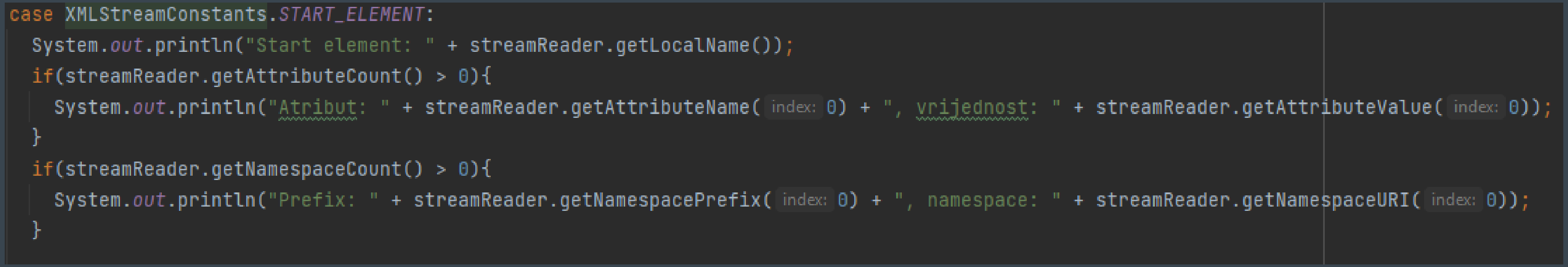
```java
startElement.getAttributes();
startElement.getNamespaces();
```



```
Comment:  Ovo je drugi komentar
Start element: User
Prefix: cc, namespace: http://test/test
Start element: id
Value of element: 2
End element: id
```



```
End element: firstName
Start element: lastName
Atribut: atribut, vrijednost: Vrijednost atributa 2
Value of element: Anić
End element: lastName
```

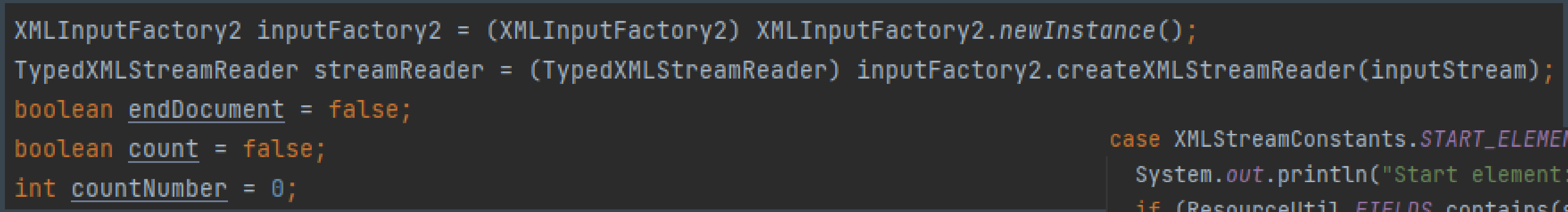


```java
case XMLStreamConstants.START_ELEMENT:
  System.out.println("Start element: " + streamReader.getLocalName());
  if(streamReader.getAttributeCount() > 0){
    System.out.println("Atribut: " + streamReader.getAttributeName( index: 0) + ", vrijednost: " + streamReader.getAttributeValue( index: 0));
  }
  if(streamReader.getNamespaceCount() > 0){
    System.out.println("Prefix: " + streamReader.getNamespacePrefix( index: 0) + ", namespace: " + streamReader.getNamespaceURI( index: 0));
  }
```

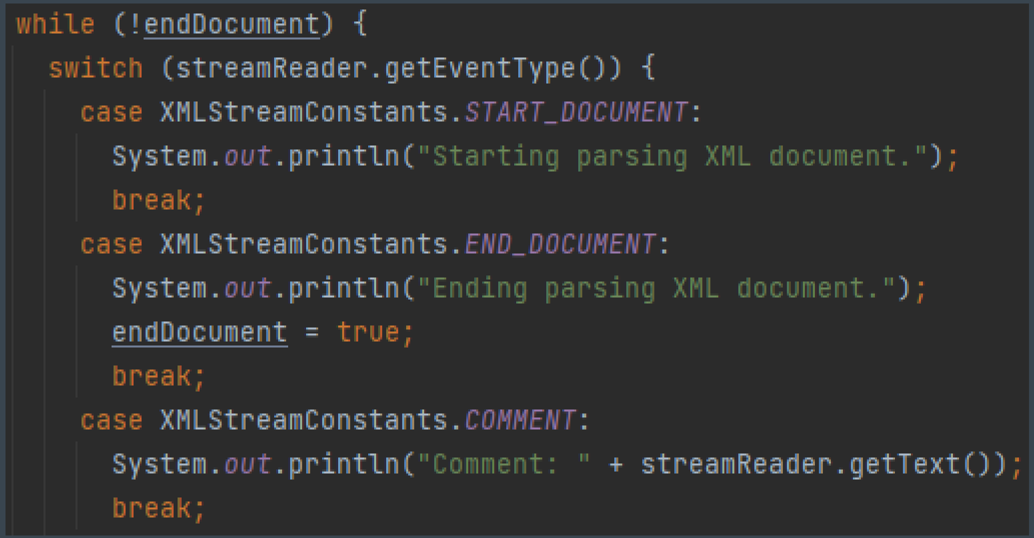
XMLStreamReader vs XMLEventReader

	1	2	3	4	5	6	7	8	9	10	avg
XMLStreamReader	131	80	88	84	89	91	96	99	139	103	**100**
XMLStreamReader (*coalescing*)	114	136	128	136	140	107	179	144	136	128	**135**
XMLEventReader	148	161	226	173	136	255	170	168	171	160	**177**

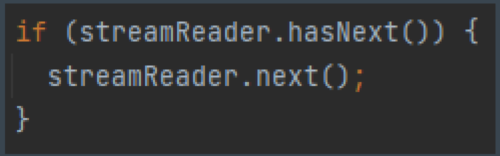
TypedXMLStreamReader



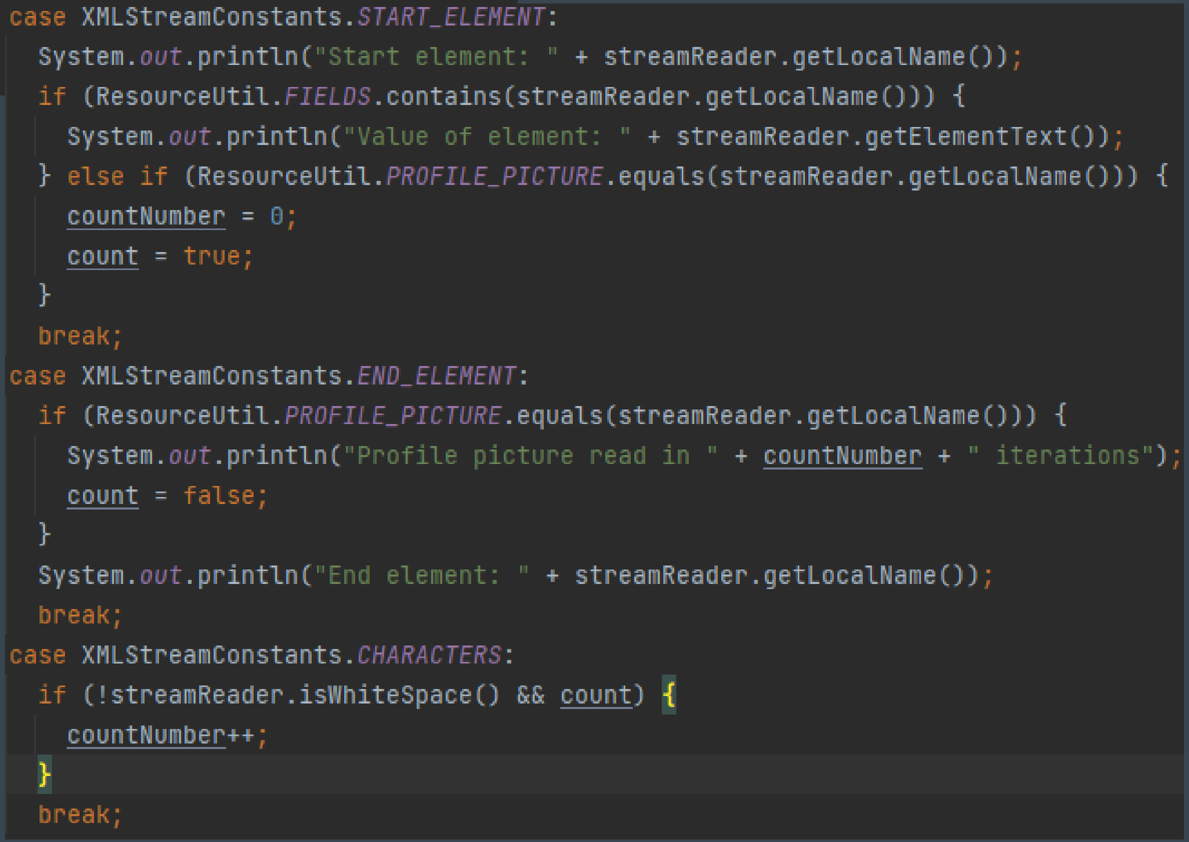
```java
XMLInputFactory2 inputFactory2 = (XMLInputFactory2) XMLInputFactory2.newInstance();
TypedXMLStreamReader streamReader = (TypedXMLStreamReader) inputFactory2.createXMLStreamReader(inputStream);
boolean endDocument = false;
boolean count = false;
int countNumber = 0;
```



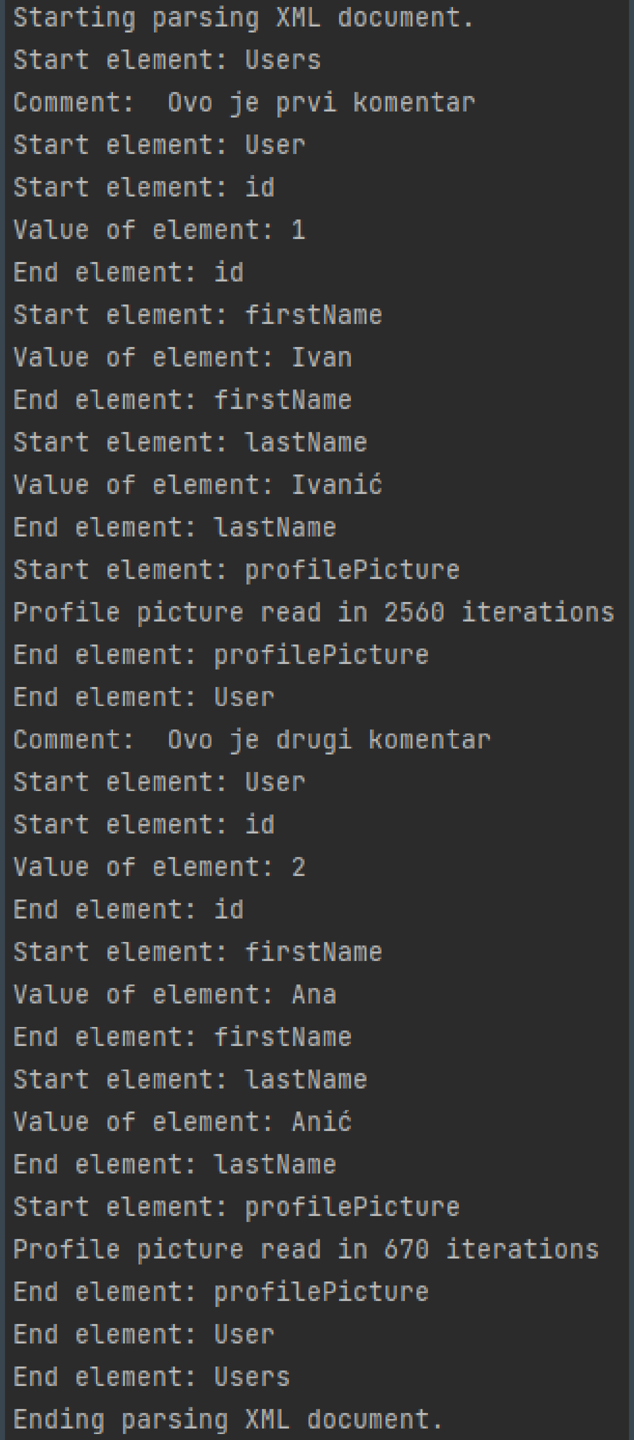
```java
while (!endDocument) {
  switch (streamReader.getEventType()) {
    case XMLStreamConstants.START_DOCUMENT:
      System.out.println("Starting parsing XML document.");
      break;
    case XMLStreamConstants.END_DOCUMENT:
      System.out.println("Ending parsing XML document.");
      endDocument = true;
      break;
    case XMLStreamConstants.COMMENT:
      System.out.println("Comment: " + streamReader.getText());
      break;
```



```java
if (streamReader.hasNext()) {
  streamReader.next();
}
```

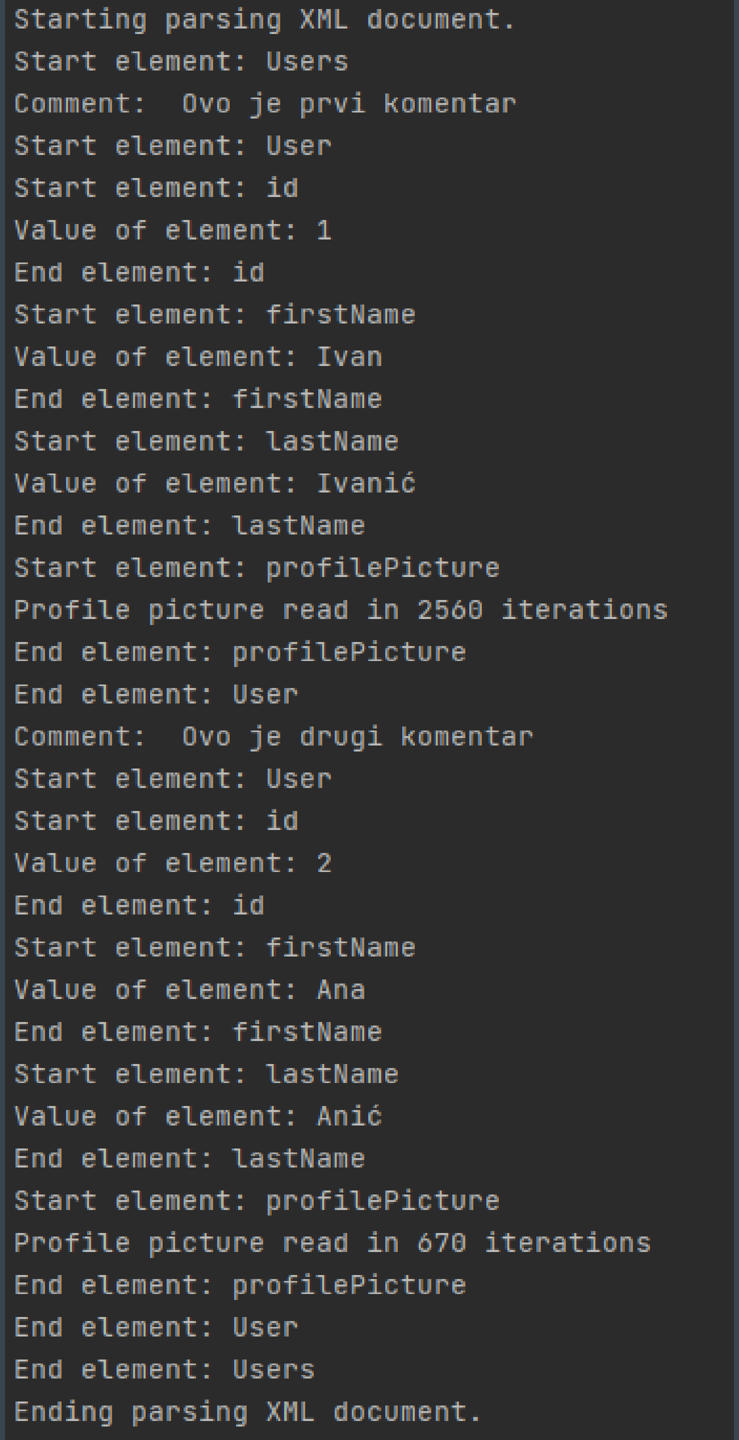


```java
case XMLStreamConstants.START_ELEMENT:
  System.out.println("Start element: " + streamReader.getLocalName());
  if (ResourceUtil.FIELDS.contains(streamReader.getLocalName())) {
    System.out.println("Value of element: " + streamReader.getElementText());
  } else if (ResourceUtil.PROFILE_PICTURE.equals(streamReader.getLocalName())) {
    countNumber = 0;
    count = true;
  }
  break;
case XMLStreamConstants.END_ELEMENT:
  if (ResourceUtil.PROFILE_PICTURE.equals(streamReader.getLocalName())) {
    System.out.println("Profile picture read in " + countNumber + " iterations");
    count = false;
  }
  System.out.println("End element: " + streamReader.getLocalName());
  break;
case XMLStreamConstants.CHARACTERS:
  if (!streamReader.isWhiteSpace() && count) {
    countNumber++;
  }
  break;
```



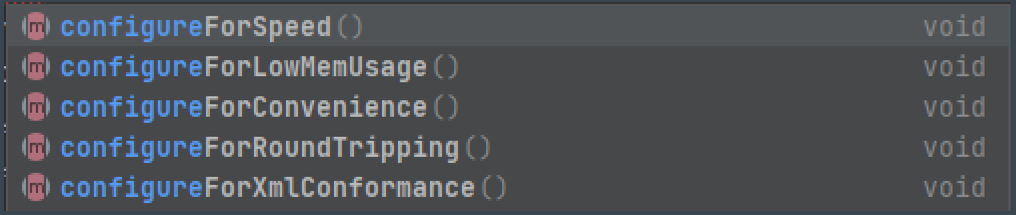
```
Starting parsing XML document.
Start element: Users
Comment:  Ovo je prvi komentar
Start element: User
Start element: id
Value of element: 1
End element: id
Start element: firstName
Value of element: Ivan
End element: firstName
Start element: lastName
Value of element: Ivanić
End element: lastName
Start element: profilePicture
Profile picture read in 2560 iterations
End element: profilePicture
End element: User
Comment:  Ovo je drugi komentar
Start element: User
Start element: id
Value of element: 2
End element: id
Start element: firstName
Value of element: Ana
End element: firstName
Start element: lastName
Value of element: Anić
End element: lastName
Start element: profilePicture
Profile picture read in 670 iterations
End element: profilePicture
End element: User
End element: Users
Ending parsing XML document.
```

==

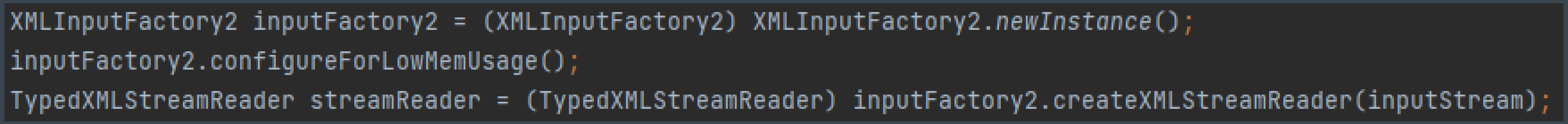


```
Starting parsing XML document.
Start element: Users
Comment:  Ovo je prvi komentar
Start element: User
Start element: id
Value of element: 1
End element: id
Start element: firstName
Value of element: Ivan
End element: firstName
Start element: lastName
Value of element: Ivanić
End element: lastName
Start element: profilePicture
Profile picture read in 2560 iterations
End element: profilePicture
End element: User
Comment:  Ovo je drugi komentar
Start element: User
Start element: id
Value of element: 2
End element: id
Start element: firstName
Value of element: Ana
End element: firstName
Start element: lastName
Value of element: Anić
End element: lastName
Start element: profilePicture
Profile picture read in 670 iterations
End element: profilePicture
End element: User
End element: Users
Ending parsing XML document.
```

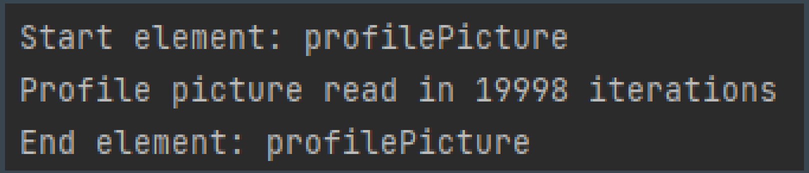
Konfiguriranje



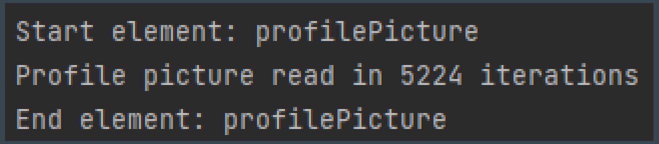
```
configureForSpeed()              void
configureForLowMemUsage()        void
configureForConvenience()        void
configureForRoundTripping()      void
configureForXmlConformance()     void
```



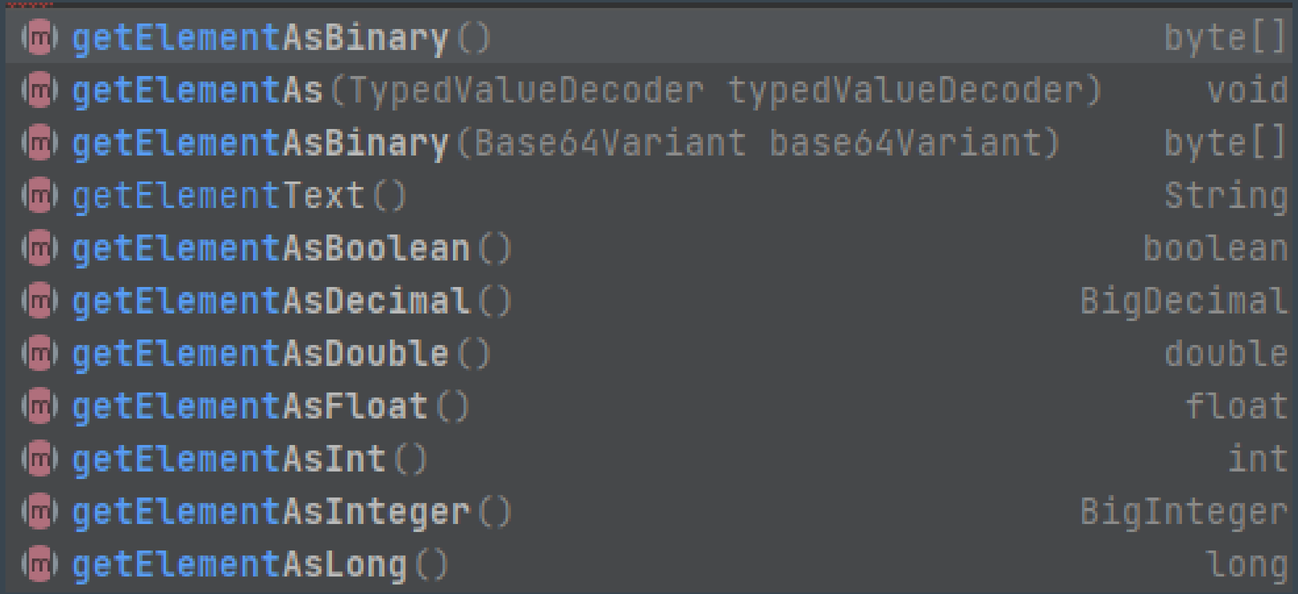
```
XMLInputFactory2 inputFactory2 = (XMLInputFactory2) XMLInputFactory2.newInstance();
inputFactory2.configureForLowMemUsage();
TypedXMLStreamReader streamReader = (TypedXMLStreamReader) inputFactory2.createXMLStreamReader(inputStream);
```



```
Start element: profilePicture
Profile picture read in 19998 iterations
End element: profilePicture
```



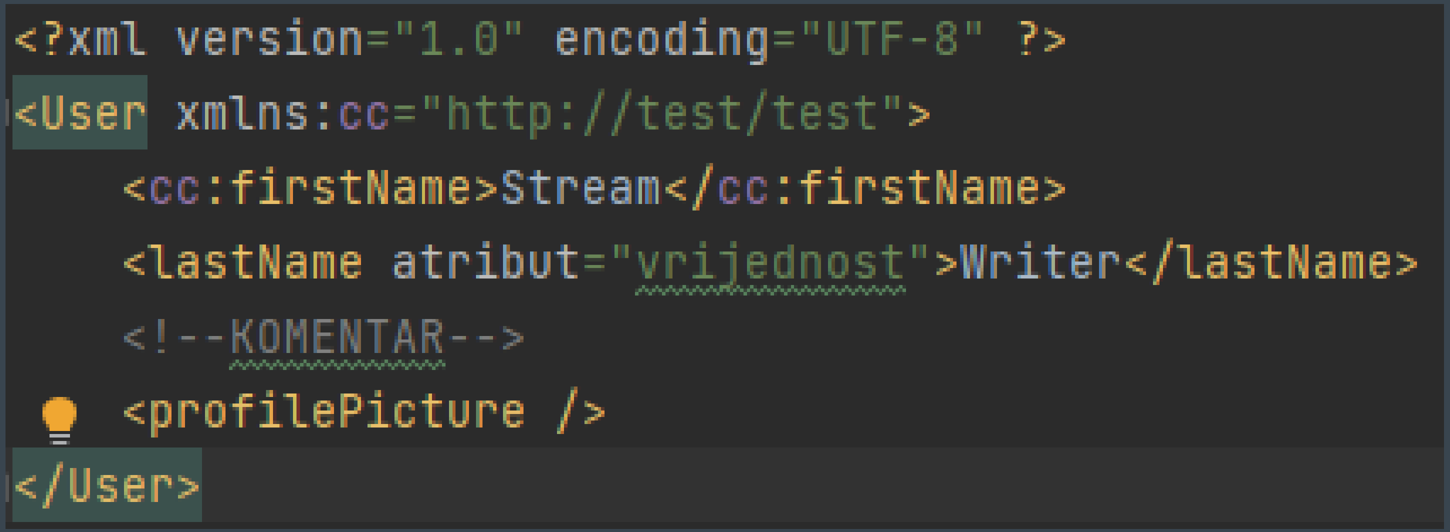
```
Start element: profilePicture
Profile picture read in 5224 iterations
End element: profilePicture
```



```
getElementAsBinary()                                         byte[]
getElementAs(TypedValueDecoder typedValueDecoder)              void
getElementAsBinary(Base64Variant base64Variant)              byte[]
getElementText()                                             String
getElementAsBoolean()                                       boolean
getElementAsDecimal()                                    BigDecimal
getElementAsDouble()                                         double
getElementAsFloat()                                           float
getElementAsInt()                                               int
getElementAsInteger()                                    BigInteger
getElementAsLong()                                             long
```

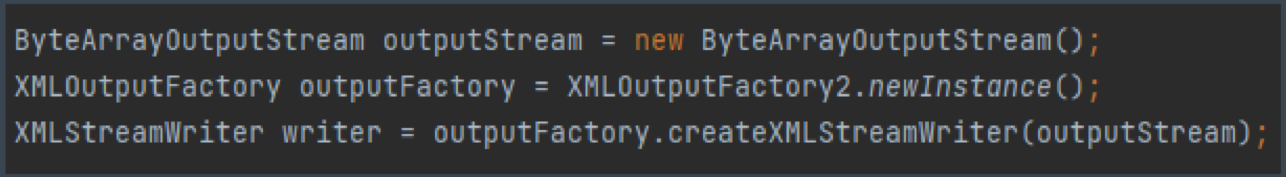
	1	2	3	4	5	6	7	8	9	10	avg
XMLStreamReader	131	80	88	84	89	91	96	99	139	103	**100**
XMLStreamReader (*coalescing*)	114	136	128	136	140	107	179	144	136	128	**135**
XMLEventReader	148	161	226	173	136	255	170	168	171	160	**177**
TypedXMLStream Reader	77	88	83	64	112	86	99	104	66	90	**87**
TypedXMLStream Reader – memory	62	64	112	96	84	96	107	128	104	104	**96**

Writing API

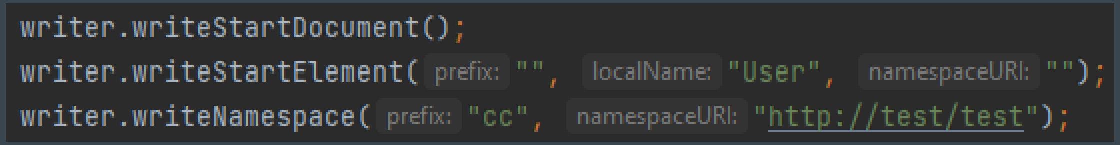


```xml
<?xml version="1.0" encoding="UTF-8" ?>
<User xmlns:cc="http://test/test">
    <cc:firstName>Stream</cc:firstName>
    <lastName atribut="vrijednost">Writer</lastName>
    <!--KOMENTAR-->
    <profilePicture />
</User>
```

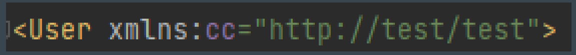
XMLStreamWriter



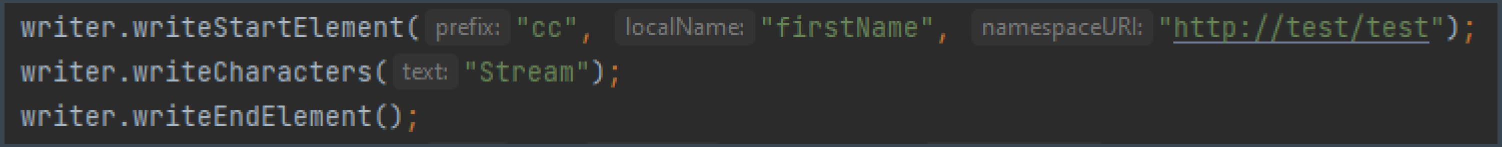
```
ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
XMLOutputFactory outputFactory = XMLOutputFactory2.newInstance();
XMLStreamWriter writer = outputFactory.createXMLStreamWriter(outputStream);
```



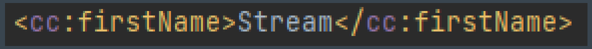
```
writer.writeStartDocument();
writer.writeStartElement( prefix: "", localName: "User", namespaceURI: "");
writer.writeNamespace( prefix: "cc", namespaceURI: "http://test/test");
```



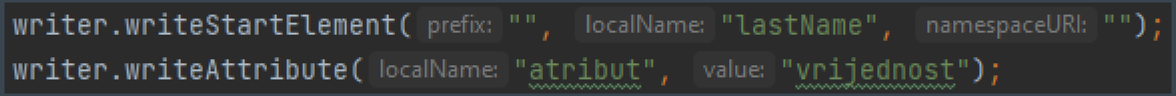
```
<User xmlns:cc="http://test/test">
```



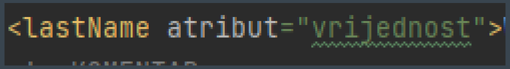
```
writer.writeStartElement( prefix: "cc", localName: "firstName", namespaceURI: "http://test/test");
writer.writeCharacters( text: "Stream");
writer.writeEndElement();
```



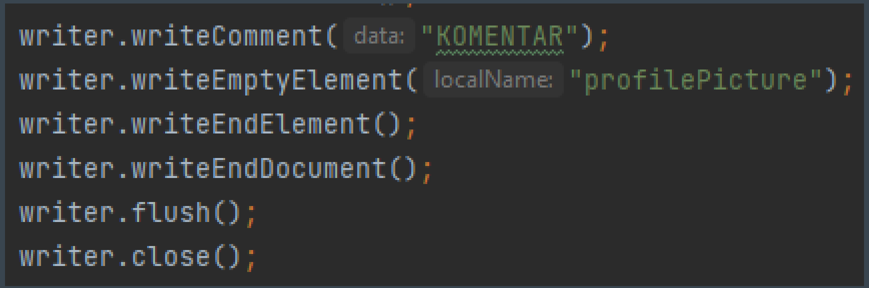
```
<cc:firstName>Stream</cc:firstName>
```



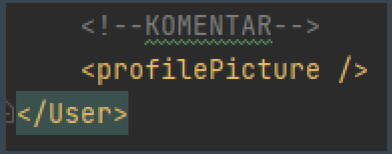
```
writer.writeStartElement( prefix: "", localName: "lastName", namespaceURI: "");
writer.writeAttribute( localName: "atribut", value: "vrijednost");
```



```
<lastName atribut="vrijednost">
```

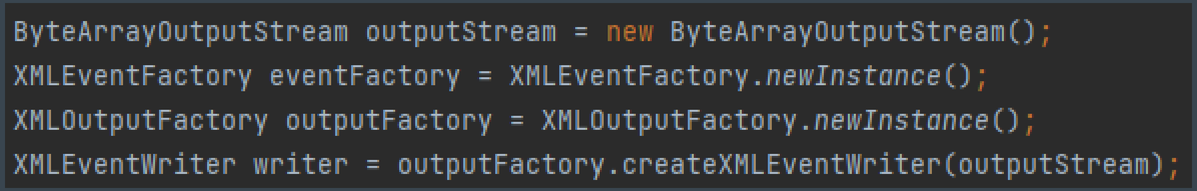


```
writer.writeComment( data: "KOMENTAR");
writer.writeEmptyElement( localName: "profilePicture");
writer.writeEndElement();
writer.writeEndDocument();
writer.flush();
writer.close();
```

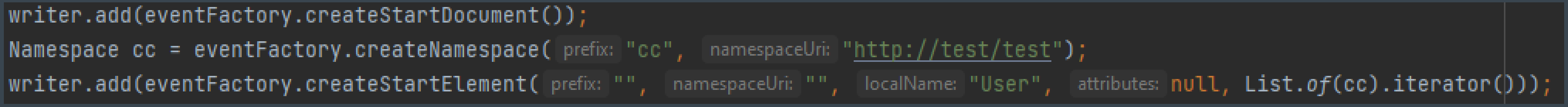


```
    <!--KOMENTAR-->
    <profilePicture />
</User>
```

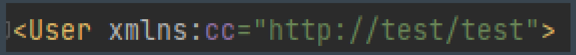
XMLEventReader



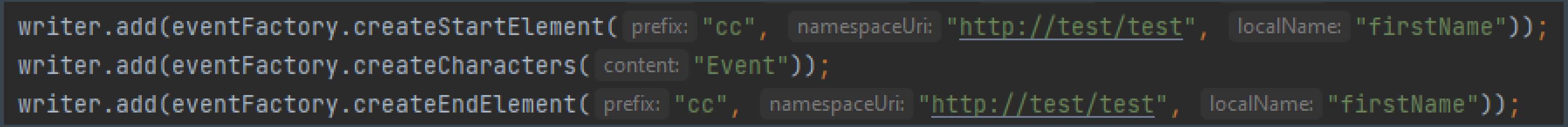
```
ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
XMLEventFactory eventFactory = XMLEventFactory.newInstance();
XMLOutputFactory outputFactory = XMLOutputFactory.newInstance();
XMLEventWriter writer = outputFactory.createXMLEventWriter(outputStream);
```



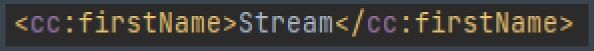
```
writer.add(eventFactory.createStartDocument());
Namespace cc = eventFactory.createNamespace( prefix: "cc", namespaceUri: "http://test/test");
writer.add(eventFactory.createStartElement( prefix: "", namespaceUri: "", localName: "User", attributes: null, List.of(cc).iterator()));
```



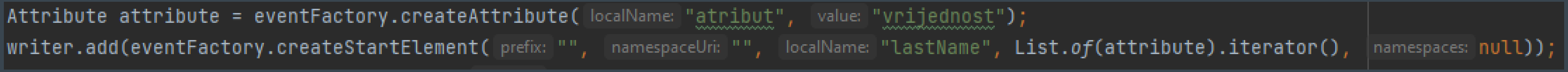
```
<User xmlns:cc="http://test/test">
```



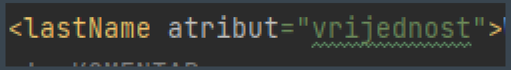
```
writer.add(eventFactory.createStartElement( prefix: "cc", namespaceUri: "http://test/test", localName: "firstName"));
writer.add(eventFactory.createCharacters( content: "Event"));
writer.add(eventFactory.createEndElement( prefix: "cc", namespaceUri: "http://test/test", localName: "firstName"));
```



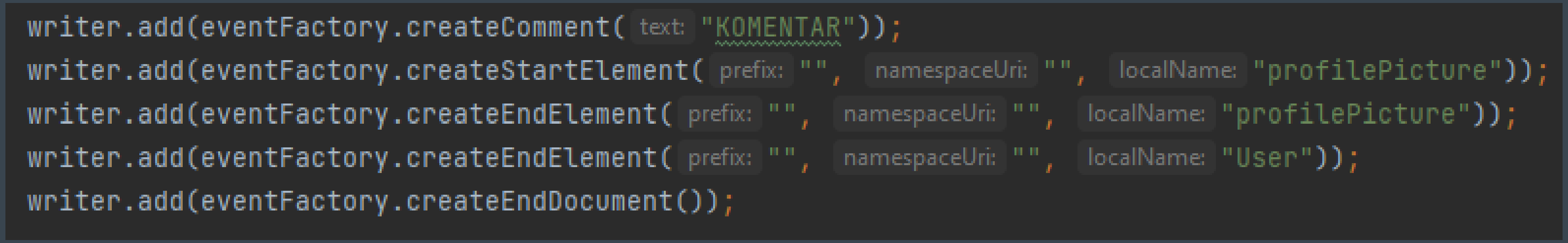
```
<cc:firstName>Stream</cc:firstName>
```



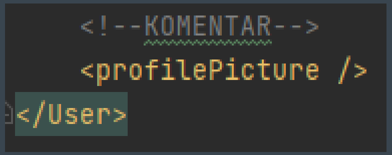
```
Attribute attribute = eventFactory.createAttribute( localName: "atribut",  value: "vrijednost");
writer.add(eventFactory.createStartElement( prefix: "",  namespaceUri: "",  localName: "lastName", List.of(attribute).iterator(),  namespaces: null));
```



```
<lastName atribut="vrijednost">
```



```
writer.add(eventFactory.createComment( text: "KOMENTAR"));
writer.add(eventFactory.createStartElement( prefix: "",  namespaceUri: "",  localName: "profilePicture"));
writer.add(eventFactory.createEndElement( prefix: "",  namespaceUri: "",  localName: "profilePicture"));
writer.add(eventFactory.createEndElement( prefix: "",  namespaceUri: "",  localName: "User"));
writer.add(eventFactory.createEndDocument());
```



```
    <!--KOMENTAR-->
    <profilePicture />
</User>
```

XMLStreamWriter vs XMLEventWriter

	1	2	3	4	5	6	7	8	9	10	avg
XMLStreamReader	160400	149300	153800	309400	156100	152500	154200	155000	151900	164200	**170680**
XMLEventReader	258500	281600	262500	272900	266300	278200	290200	265500	403700	248400	**282780**

Završne misli…

Hvala na pažnji!

CROZ