Reference Manual

Generated by Doxygen 1.2.8.1

Thu Sep 12 20:07:44 2002

Contents

Chapter 1

Package List

1.1 Package List

Here are the packages with brief descriptions (if available):

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Compound Index

3.1 Compound List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Package Documentation

5.1 Package ajc

Interfaces

- interface Enactable
- interface Scheduling
- interface Storing

Classes

- class AbstractEnactable
- class Exceptional
- class Failed
- class Schedule
- class Store
- class TaggedBuffers

5.2 Package ajc.data

Interfaces

- interface Action
- interface Agent
- interface Bindable
- interface Bindings
- interface Bool
- interface Cell
- interface Data
- interface DataConst
- interface DataFactory
- interface DataValue
- interface Datum
- interface Empty
- interface Int
- interface List
- interface Message
- interface MessageTag
- interface Storable
- interface Text
- interface Token

5.3 Package ajc.data.pure

Classes

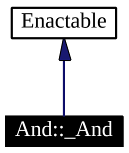
- class And::_And
- class AndExceptionally::_AndExceptionally
- class AndThen::_AndThen
- class Exceptionally::_Exceptionally
- class Hence::_Hence
- class Indivisibly::_Indivisibly
- class Otherwise::_Otherwise
- class Provide::_Provide
- class Then::_Then
- class AbstractInfixCombinator
- class AbstractPrefixCombinator
- class ActionImpl
- class AgentImpl
- class And
- class AndExceptionally
- class AndThen
- class BindingsImpl
- class BoolImpl
- class CellImpl
- class DataConstImpl
- class DataImpl
- class DataValueImpl
- class DatumImpl
- class EmptyImpl
- class Exceptionally
- class Hence
- class Indivisibly
- class IntImpl
- class ListImpl
- class MessageTagImpl
- class Otherwise
- class Provide
- class PureFactory
- class TextImpl
- class Then
- class TokenImpl

Chapter 6

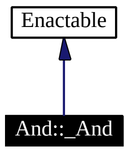
Class Documentation

6.1 And::_And Class Reference

Inheritance diagram for And::_And:



Collaboration diagram for And::_And:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.1.1 Member Function Documentation

6.1.1.1 Data And::_And::enact (Data *data*, Bindings *bindings*) [inline]

Reimplemented from Enactable.

Definition at line 106 of file ActionImpl.java.

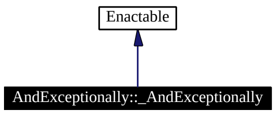
```
                                                                            {
             Data d1 = enactable1.enact(data, bindings);
             Data d2 = enactable2.enact(data, bindings);
             return d1.concat(d2);
         }
```

The documentation for this class was generated from the following file:

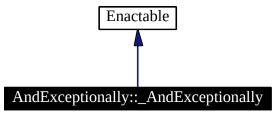
- ActionImpl.java

6.2 AndExceptionally::_AndExceptionally Class Reference

Inheritance diagram for AndExceptionally::_AndExceptionally:



Collaboration diagram for AndExceptionally::_AndExceptionally:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.2.1 Member Function Documentation

6.2.1.1 Data AndExceptionally::_AndExceptionally::enact (Data *data*, Bindings *bindings*) [inline]

Reimplemented from Enactable.

Definition at line 138 of file ActionImpl.java.

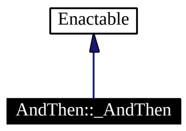
```
                                                                                                   {
             try {
                 return enactable1.enact(data,bindings);
             }
             catch (Exceptional e1) {
                 Data d1 = e1.getData();
                 try {
                     return enactable2.enact(data, bindings);
                 }
                 catch (Exceptional e2) {
                     Data d2 = e2.getData();
                     throw new Exceptional(d1.concat(d2));
                 }
             }
         }
```

The documentation for this class was generated from the following file:

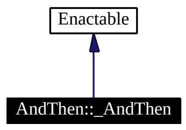
- ActionImpl.java

6.3 AndThen::_AndThen Class Reference

Inheritance diagram for AndThen::_AndThen:



Collaboration diagram for AndThen::_AndThen:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.3.1 Member Function Documentation

6.3.1.1 Data AndThen::_AndThen::enact (Data *data*, Bindings *bindings*) `[inline]`

Reimplemented from Enactable.

Definition at line 92 of file ActionImpl.java.

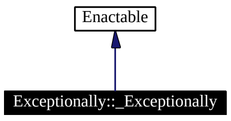
```
                                                                                  {
              Data d1 = enactable1.enact(data, bindings);
              Data d2 = enactable2.enact(data, bindings);
              return d1.concat(d2);
          }
```

The documentation for this class was generated from the following file:

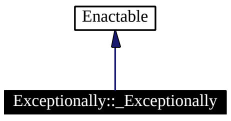
- ActionImpl.java

6.4 Exceptionally::_Exceptionally Class Reference

Inheritance diagram for Exceptionally::_Exceptionally:



Collaboration diagram for Exceptionally::_Exceptionally:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.4.1 Member Function Documentation

6.4.1.1 Data Exceptionally::_Exceptionally::enact (Data *data*, Bindings *bindings*) [inline]

Reimplemented from Enactable.

Definition at line 120 of file ActionImpl.java.

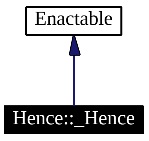
```
                                                                                   {
             try {
                 return enactable1.enact(data,bindings);
             }
             catch (Exceptional e) {
                 Data d = e.getData();
                 return enactable2.enact(d, bindings);
             }
         }
```

The documentation for this class was generated from the following file:

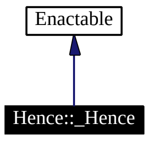
- ActionImpl.java

6.5 Hence::_Hence Class Reference

Inheritance diagram for Hence::_Hence:



Collaboration diagram for Hence::_Hence:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.5.1 Member Function Documentation

6.5.1.1 Data Hence::_Hence::enact (Data *data*, Bindings *bindings*) `[inline]`

Reimplemented from Enactable.

Definition at line 179 of file ActionImpl.java.

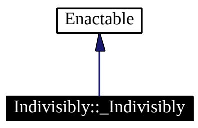
```
                                                                                  {
             return enactable2.enact(data, (Bindings)enactable1.enact(data, bindings));
         }
```

The documentation for this class was generated from the following file:

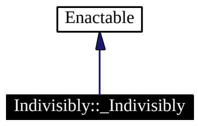
- ActionImpl.java

6.6 Indivisibly::_Indivisibly Class Reference

Inheritance diagram for Indivisibly::_Indivisibly:



Collaboration diagram for Indivisibly::_Indivisibly:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.6.1 Member Function Documentation

6.6.1.1 Data Indivisibly::_Indivisibly::enact (Data *data*, Bindings *bindings*) `[inline]`

Reimplemented from Enactable.

Definition at line 192 of file ActionImpl.java.

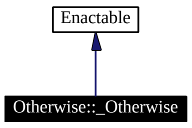
```
00192                                                                          {
00193             synchronized (Store.getStore()) {
00194                 return enactable1.enact(data, bindings);
00195             }
00196         }
```

The documentation for this class was generated from the following file:

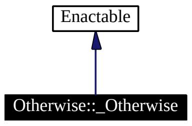
- ActionImpl.java

6.7 Otherwise::_Otherwise Class Reference

Inheritance diagram for Otherwise::_Otherwise:



Collaboration diagram for Otherwise::_Otherwise:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.7.1 Member Function Documentation

6.7.1.1 Data Otherwise::_Otherwise::enact (Data *data*, Bindings *bindings*) [inline]

Reimplemented from Enactable.

Definition at line 162 of file ActionImpl.java.

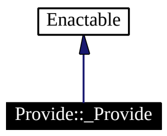
```
00162                                                                                  {
00163             try {
00164                 return enactable1.enact(data, bindings);
00165             }
00166             catch (Failed f) {
00167                 return enactable2.enact(data, bindings);
00168             }
00169         }
```

The documentation for this class was generated from the following file:

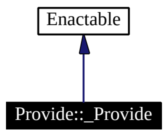
- ActionImpl.java

6.8 Provide::_Provide Class Reference

Inheritance diagram for Provide::_Provide:



Collaboration diagram for Provide::_Provide:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.8.1 Member Function Documentation

6.8.1.1 Data Provide::_Provide::enact (Data *data*, Bindings *bindings*) `[inline]`

Reimplemented from Enactable.

Definition at line 55 of file DataImpl.java.

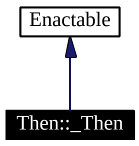
```
00055                                                                                         {
00056                 return provided;
00057             }
```

The documentation for this class was generated from the following file:

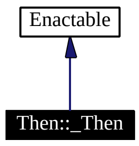
- DataImpl.java

6.9 Then::_Then Class Reference

Inheritance diagram for Then::_Then:



Collaboration diagram for Then::_Then:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.9.1 Member Function Documentation

6.9.1.1 Data Then::_Then::enact (Data *data*, Bindings *bindings*) [inline]

Reimplemented from Enactable.

Definition at line 80 of file ActionImpl.java.

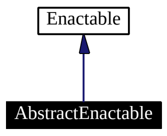
```
                                                                              {
             return enactable2.enact(enactable1.enact(data, bindings), bindings);
        }
```

The documentation for this class was generated from the following file:

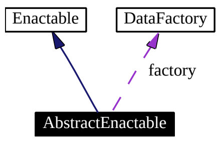
- ActionImpl.java

6.10 AbstractEnactable Class Reference

Inheritance diagram for AbstractEnactable:



Collaboration diagram for AbstractEnactable:



Protected Methods

- Data fail () throws Failed
- Data raise (Data data) throws Exceptional
- Data catchExceptional (Exception e) throws Exception
- void catchFailed (Exception e) throws Exception

Protected Attributes

- DataFactory factory

6.10.1 Member Function Documentation

6.10.1.1 Data AbstractEnactable::catchExceptional (Exception *e*) `[inline, protected]`

Definition at line 21 of file AbstractEnactable.java.

```
                                                                  {
         if (e instanceof Exceptional)
             return ((Exceptional)e).getData();
         if (e instanceof Failed)
             throw (Failed)e;
         if (e instanceof ClassCastException)
             return factory.makeEmpty();
         throw e;
     }
```

6.10.1.2 void AbstractEnactable::catchFailed (Exception *e*) `[inline, protected]`

Definition at line 31 of file AbstractEnactable.java.

```
                                                             {
         if (!(e instanceof Failed))
             throw e;
     }
```

6.10.1.3 Data AbstractEnactable::fail () [inline, protected]

Definition at line 14 of file AbstractEnactable.java.

```
00014                                         {
00015          throw new Failed();
00016      }
```

6.10.1.4 Data AbstractEnactable::raise (Data *data*) [inline, protected]

Definition at line 17 of file AbstractEnactable.java.

```
00017                                                        {
00018          throw new Exceptional(data);
00019      }
```

6.10.2 Member Data Documentation

6.10.2.1 DataFactory AbstractEnactable::factory [protected]

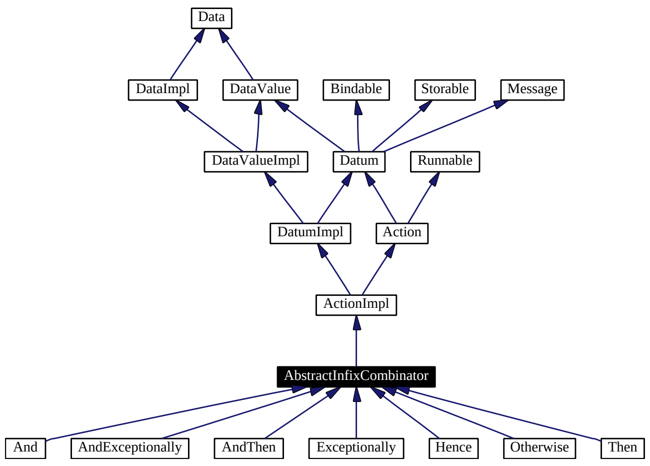
Definition at line 12 of file AbstractEnactable.java.

The documentation for this class was generated from the following file:

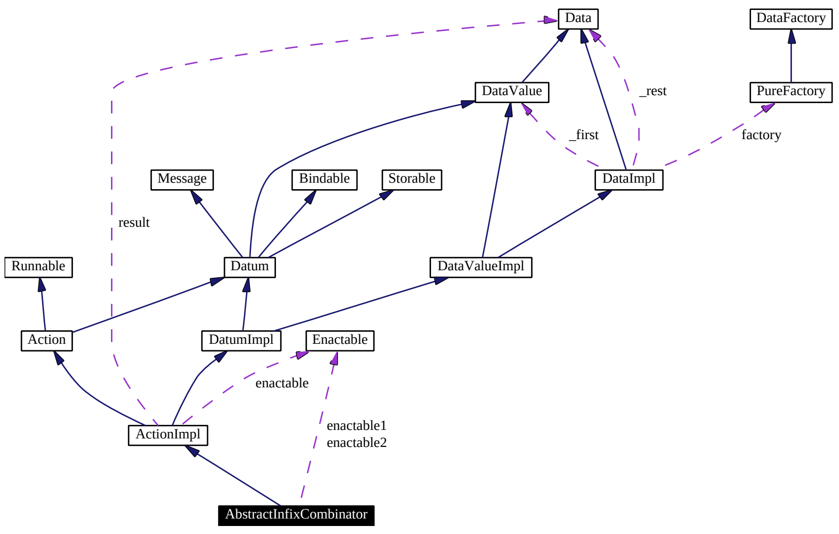
- AbstractEnactable.java

6.11 AbstractInfixCombinator Class Reference

Inheritance diagram for AbstractInfixCombinator:



Collaboration diagram for AbstractInfixCombinator:



Private Methods

- AbstractInfixCombinator (PureFactory factory, Enactable enactable1, Enactable enactable2)

Private Attributes

- Enactable enactable1
- Enactable enactable2

6.11.1 Constructor & Destructor Documentation

6.11.1.1 AbstractInfixCombinator::AbstractInfixCombinator (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) [inline, private]

Definition at line 10 of file AbstractInfixCombinator.java.

```
                                              {
         super(factory);
         this.enactable1 = enactable1;
         this.enactable2 = enactable2;
     }
```

6.11.2 Member Data Documentation

6.11.2.1 Enactable AbstractInfixCombinator::enactable1 [private]

Definition at line 8 of file AbstractInfixCombinator.java.

6.11.2.2 Enactable AbstractInfixCombinator::enactable2 [private]

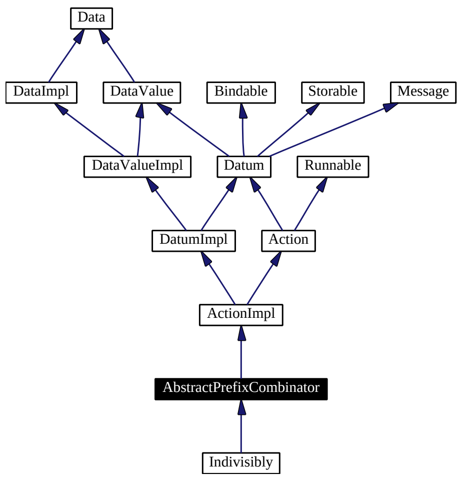
Definition at line 9 of file AbstractInfixCombinator.java.

The documentation for this class was generated from the following file:

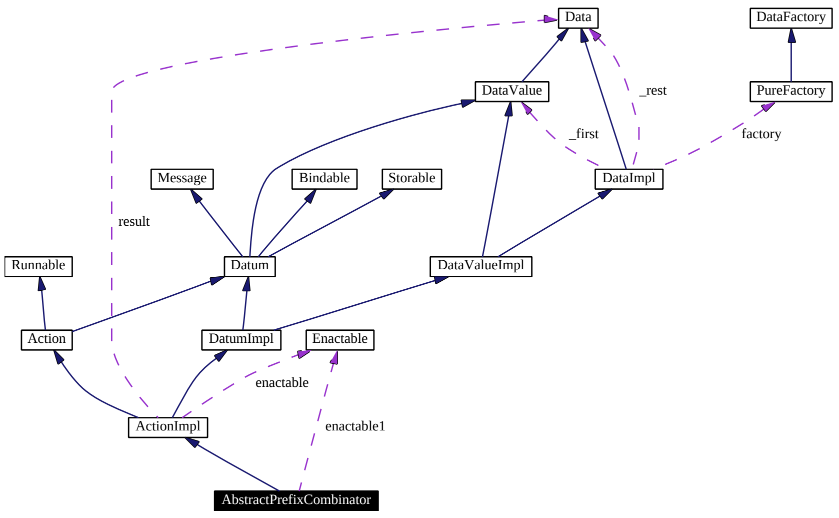
- AbstractInfixCombinator.java

6.12 AbstractPrefixCombinator Class Reference

Inheritance diagram for AbstractPrefixCombinator:



Collaboration diagram for AbstractPrefixCombinator:



Protected Attributes

- Enactable enactable1

Private Methods

- AbstractPrefixCombinator (PureFactory factory, Enactable enactable1)

6.12.1 Constructor & Destructor Documentation

6.12.1.1 AbstractPrefixCombinator::AbstractPrefixCombinator (PureFactory *factory*, Enactable *enactable1*) `[inline, private]`

Definition at line 9 of file AbstractPrefixCombinator.java.

```
                                                                         {
              super(factory);
              this.enactable1 = enactable1;
          }
```

6.12.2 Member Data Documentation

6.12.2.1 Enactable AbstractPrefixCombinator::enactable1 `[protected]`

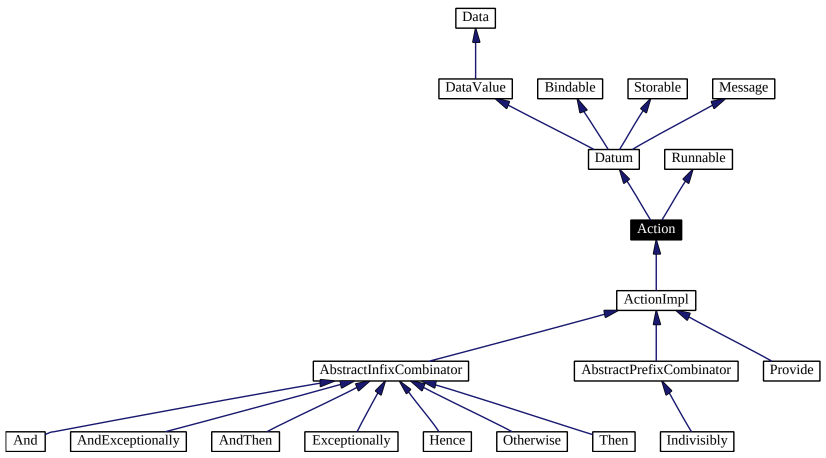
Definition at line 8 of file AbstractPrefixCombinator.java.

The documentation for this class was generated from the following file:

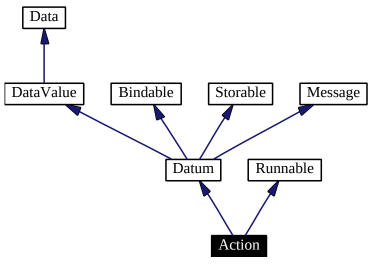
- AbstractPrefixCombinator.java

6.13 Action Interface Reference

Inheritance diagram for Action:



Collaboration diagram for Action:



Public Methods

- Enactable enactableValue ()
- Data enact () throws Exceptional, Failed
- Action then (Action action)
- Action andThen (Action action)
- Action and (Action action)
- Action exceptionally (Action action)
- Action andExceptionally (Action action)
- Action otherwise (Action action)
- Action hence (Action action)
- Action indivisibly ()

6.13.1 Member Function Documentation

6.13.1.1 Action Action::and (Action *action*)

Reimplemented in ActionImpl.

6.13.1.2 Action Action::andExceptionally (Action *action*)

Reimplemented in ActionImpl.

6.13.1.3 Action Action::andThen (Action *action*)

Reimplemented in ActionImpl.

6.13.1.4 Data Action::enact ()

Reimplemented in ActionImpl, and ActionImpl.

Referenced by ActionImpl::run().

6.13.1.5 Enactable Action::enactableValue ()

Reimplemented in ActionImpl.

Referenced by ActionImpl::and(), ActionImpl::andExceptionally(), ActionImpl::andThen(), ActionImpl::exceptionally(), ActionImpl::hence(), ActionImpl::otherwise(), and ActionImpl::then().

6.13.1.6 Action Action::exceptionally (Action *action*)

Reimplemented in ActionImpl.

6.13.1.7 Action Action::hence (Action *action*)

Reimplemented in ActionImpl.

6.13.1.8 Action Action::indivisibly ()

Reimplemented in ActionImpl.

6.13.1.9 Action Action::otherwise (Action *action*)

Reimplemented in ActionImpl.

6.13.1.10 Action Action::then (Action *action*)

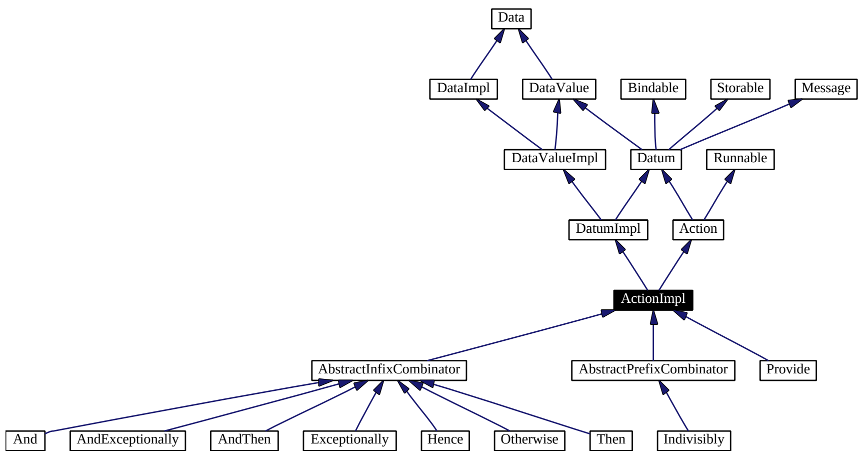
Reimplemented in ActionImpl.

The documentation for this interface was generated from the following file:

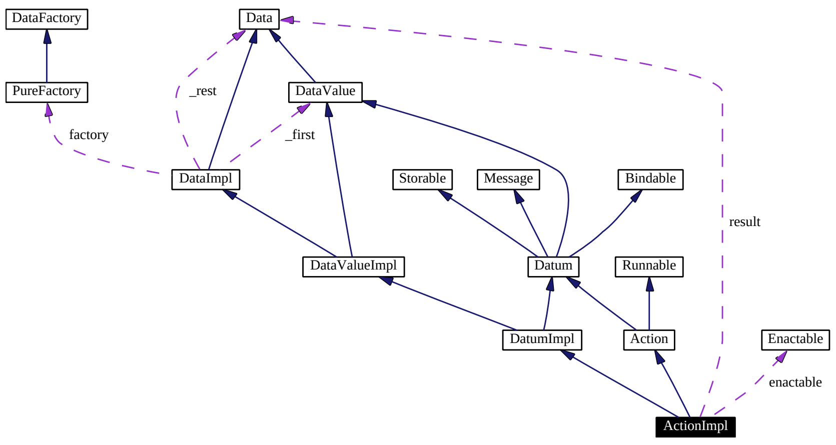
- Action.java

6.14 ActionImpl Class Reference

Inheritance diagram for ActionImpl:



Collaboration diagram for ActionImpl:



Public Methods

- Enactable getEnactable ()
- Data enact ()
- Action giveThen (Action action)
- Action giveAndThen (Action action)
- Action giveAnd (Action action)
- Action giveExceptionally (Action action)
- Action giveAndExceptionally (Action action)
- Action giveOtherwise (Action action)
- Action giveHence (Action action)
- Action giveIndivisibly ()
- Enactable enactableValue ()
- Data enact () throws Exceptional, Failed

- Action then (Action action)
- Action andThen (Action action)
- Action and (Action action)
- Action exceptionally (Action action)
- Action andExceptionally (Action action)
- Action otherwise (Action action)
- Action hence (Action action)
- Action indivisibly ()
- void run ()
- Empty equals (Data data) throws Exceptional

Private Methods

- ActionImpl (PureFactory factory, Enactable enactable)
- ActionImpl (PureFactory factory)
- ActionImpl (PureFactory factory, Enactable enactable)
- ActionImpl (PureFactory factory)

Private Attributes

- Data result
- Enactable enactable

6.14.1 Constructor & Destructor Documentation

6.14.1.1 ActionImpl::ActionImpl (PureFactory *factory*, Enactable *enactable*) `[inline, private]`

Definition at line 17 of file AbstractAction.java.

```
00017                                                           {
00018         super(factory);
00019         this.enactable = enactable;
00020     }
```

6.14.1.2 ActionImpl::ActionImpl (PureFactory *factory*) `[inline, private]`

Definition at line 22 of file AbstractAction.java.

```
00022                                       {
00023         this(factory, null);
00024     }
```

6.14.1.3 ActionImpl::ActionImpl (PureFactory *factory*, Enactable *enactable*) `[inline, private]`

Definition at line 13 of file ActionImpl.java.

```
00013                                                           {
00014         super(factory);
00015         this.enactable = enactable;
00016     }
```

6.14.1.4 ActionImpl::ActionImpl (PureFactory *factory*) [inline, private]

Definition at line 18 of file ActionImpl.java.

```
                                    {
        this(factory, null);
    }
```

6.14.2 Member Function Documentation

6.14.2.1 Action ActionImpl::and (Action *action*) [inline]

Reimplemented from Action.

Definition at line 38 of file ActionImpl.java.

```
                                      {
        return new And(factory, enactable, action.enactableValue());
    }
```

6.14.2.2 Action ActionImpl::andExceptionally (Action *action*) [inline]

Reimplemented from Action.

Definition at line 44 of file ActionImpl.java.

```
                                                   {
        return new AndExceptionally(factory, enactable, action.enactableValue());
    }
```

6.14.2.3 Action ActionImpl::andThen (Action *action*) [inline]

Reimplemented from Action.

Definition at line 35 of file ActionImpl.java.

```
                                          {
        return new AndThen(factory, enactable, action.enactableValue());
    }
```

6.14.2.4 Data ActionImpl::enact () [inline]

Reimplemented from Action.

Definition at line 28 of file ActionImpl.java.

```
                                                    {
        return enactable.enact(factory.makeEmpty(), factory.makeNoBindings());
    }
```

6.14.2.5 Data ActionImpl::enact () [inline]

Reimplemented from Action.

Definition at line 32 of file AbstractAction.java.

```
                         {
        return enactable.enact(factory.makeEmpty(), factory.makeNoBindings());
    }
```

6.14.2.6 Enactable ActionImpl::enactableValue () [inline]

Reimplemented from Action.

Definition at line 24 of file ActionImpl.java.

```
                                       {
        return enactable;
    }
```

6.14.2.7 Empty ActionImpl::equals (Data *data*) [inline]

Reimplemented from DataValueImpl.

Definition at line 68 of file ActionImpl.java.

```
                                                       {
        if (data instanceof Action)
            if (enactable == ((Action)data).enactableValue())
                return factory.makeEmpty();
        throw new Exceptional(factory.makeEmpty());
    }
```

6.14.2.8 Action ActionImpl::exceptionally (Action *action*) [inline]

Reimplemented from Action.

Definition at line 41 of file ActionImpl.java.

```
                                                {
        return new Exceptionally(factory, enactable, action.enactableValue());
    }
```

6.14.2.9 Enactable ActionImpl::getEnactable () [inline]

Definition at line 28 of file AbstractAction.java.

```
                                     {
        return enactable;
    }
```

6.14.2.10 Action ActionImpl::giveAnd (Action *action*) [inline]

Definition at line 42 of file AbstractAction.java.

```
00042                                           {
00043           return new And(factory, enactable, action.getEnactable());
00044       }
```

6.14.2.11 Action ActionImpl::giveAndExceptionally (Action *action*) [inline]

Definition at line 48 of file AbstractAction.java.

```
00048                                                        {
00049           return new AndExceptionally(factory, enactable, action.getEnactable());
00050       }
```

6.14.2.12 Action ActionImpl::giveAndThen (Action *action*) [inline]

Definition at line 39 of file AbstractAction.java.

```
00039                                               {
00040           return new AndThen(factory, enactable, action.getEnactable());
00041       }
```

6.14.2.13 Action ActionImpl::giveExceptionally (Action *action*) [inline]

Definition at line 45 of file AbstractAction.java.

```
00045                                                     {
00046           return new Exceptionally(factory, enactable, action.getEnactable());
00047       }
```

6.14.2.14 Action ActionImpl::giveHence (Action *action*) [inline]

Definition at line 54 of file AbstractAction.java.

```
00054                                             {
00055           return new Hence(factory, enactable, action.getEnactable());
00056       }
```

6.14.2.15 Action ActionImpl::giveIndivisibly () [inline]

Definition at line 57 of file AbstractAction.java.

```
00057                                      {
00058           return new Indivisibly(factory, enactable);
00059       }
```

6.14.2.16 Action ActionImpl::giveOtherwise (Action *action*) [inline]

Definition at line 51 of file AbstractAction.java.

```
00051                                             {
00052         return new AndOtherwise(factory, enactable, action.getEnactable());
00053     }
```

6.14.2.17 Action ActionImpl::giveThen (Action *action*) [inline]

Definition at line 36 of file AbstractAction.java.

```
00036                                        {
00037         return new Then(factory, enactable, action.getEnactable());
00038     }
```

6.14.2.18 Action ActionImpl::hence (Action *action*) [inline]

Reimplemented from Action.

Definition at line 50 of file ActionImpl.java.

```
00050                                     {
00051         return new Hence(factory, enactable, action.enactableValue());
00052     }
```

6.14.2.19 Action ActionImpl::indivisibly () [inline]

Reimplemented from Action.

Definition at line 53 of file ActionImpl.java.

```
00053                              {
00054         return new Indivisibly(factory, enactable);
00055     }
```

6.14.2.20 Action ActionImpl::otherwise (Action *action*) [inline]

Reimplemented from Action.

Definition at line 47 of file ActionImpl.java.

```
00047                                         {
00048         return new Otherwise(factory, enactable, action.enactableValue());
00049     }
```

6.14.2.21 void ActionImpl::run () [inline]

Definition at line 58 of file ActionImpl.java.

```
                       {
        try {
            Data result = enact();
        }
        catch (Exception e) {
            System.out.println("Agent exited abnormally.");
        }
    }
```

6.14.2.22 Action ActionImpl::then (Action *action*) `[inline]`

Reimplemented from Action.

Definition at line 32 of file ActionImpl.java.

```
                                       {
        return new Then(factory, enactable, action.enactableValue());
    }
```

6.14.3 Member Data Documentation

6.14.3.1 Enactable ActionImpl::enactable `[private]`

Definition at line 9 of file ActionImpl.java.

6.14.3.2 Data ActionImpl::result `[private]`

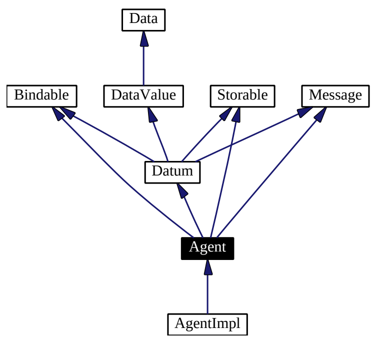
Definition at line 12 of file AbstractAction.java.

The documentation for this class was generated from the following files:

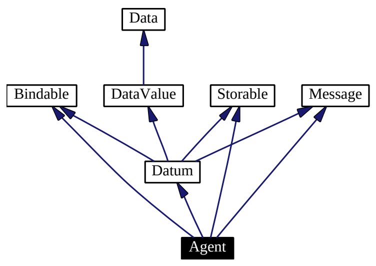
- AbstractAction.java
- ActionImpl.java

6.15 Agent Interface Reference

Inheritance diagram for Agent:



Collaboration diagram for Agent:

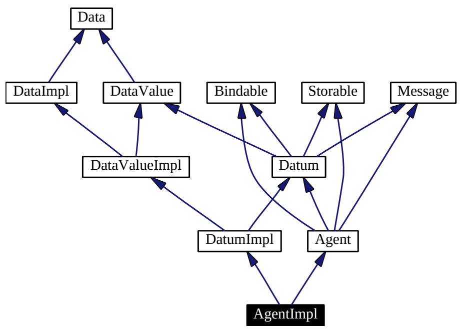


The documentation for this interface was generated from the following file:

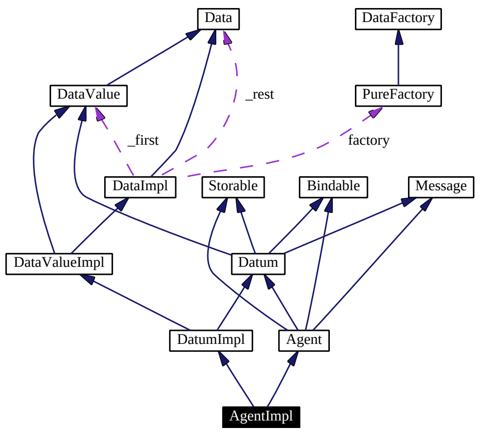
- Agent.java

6.16 AgentImpl Class Reference

Inheritance diagram for AgentImpl:



Collaboration diagram for AgentImpl:



Private Methods

- AgentImpl (PureFactory factory)

Private Attributes

- int id

Static Private Attributes

- int agents = 0

6.16.1 Constructor & Destructor Documentation

6.16.1.1 AgentImpl::AgentImpl (PureFactory *factory*) `[inline, private]`

Definition at line 15 of file AgentImpl.java.

```
                                   {
        super(factory);
        id = agents++;
    }
```

6.16.2 Member Data Documentation

6.16.2.1 int AgentImpl::agents = 0 `[static, private]`

Definition at line 9 of file AgentImpl.java.

6.16.2.2 int AgentImpl::id `[private]`

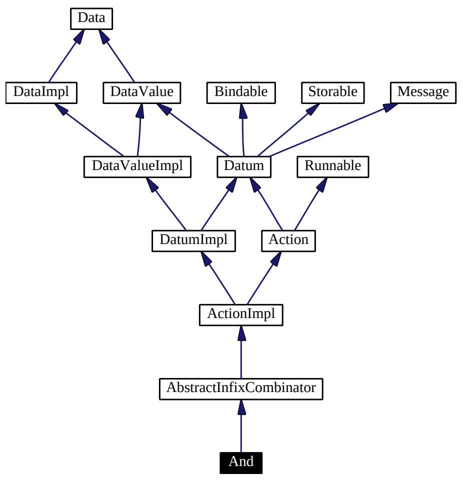
Definition at line 13 of file AgentImpl.java.

The documentation for this class was generated from the following file:

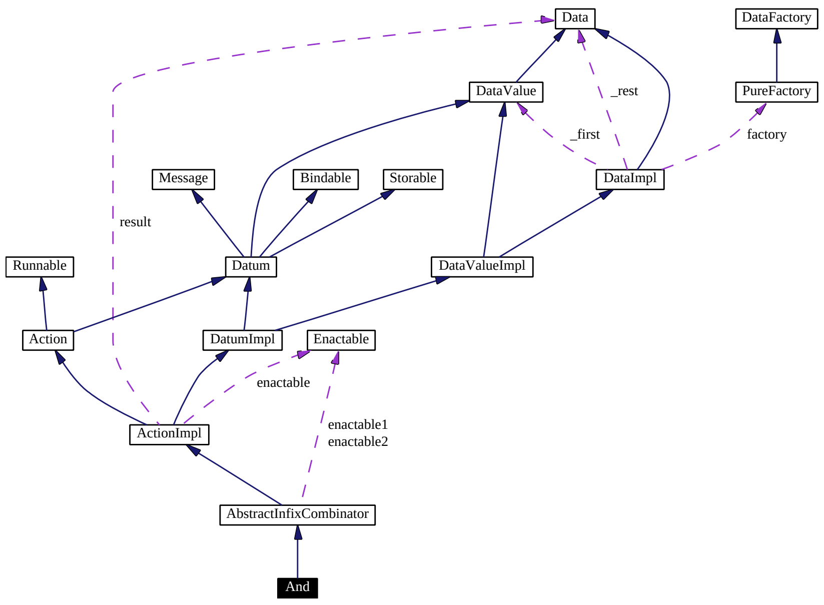
- AgentImpl.java

6.17 And Class Reference

Inheritance diagram for And:



Collaboration diagram for And:



Public Methods

- And (PureFactory factory, Enactable enactable1, Enactable enactable2)
- Data enact (Data data, Bindings bindings) throws Exceptional, Failed
- And (PureFactory factory, Enactable enactable1, Enactable enactable2)

6.17.1 Constructor & Destructor Documentation

6.17.1.1 And::And (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 84 of file AbstractAction.java.

```
                                                                                {
              super(factory, enactable1, enactable2);
          }
```

6.17.1.2 And::And (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 112 of file ActionImpl.java.

```
                                                                                {
              super(factory, enactable1, enactable2);
              enactable = new _And();
          }
```

6.17.2 Member Function Documentation

6.17.2.1 Data And::enact (Data *data*, Bindings *bindings*) `[inline]`

Definition at line 87 of file AbstractAction.java.

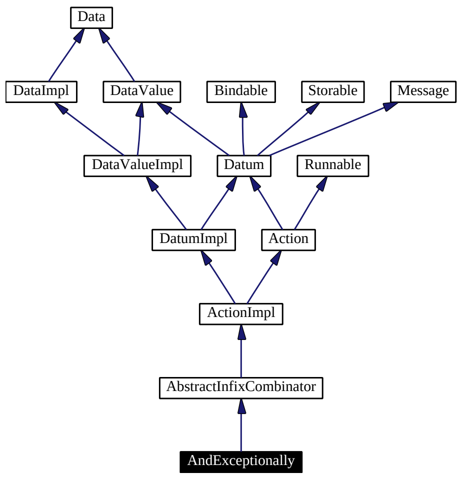
```
                                                                               {
              Data d1 = enactable1.enact(data, bindings);
              Data d2 = enactable2.enact(data, bindings);
              return d1.merge(d2);
          }
```

The documentation for this class was generated from the following files:

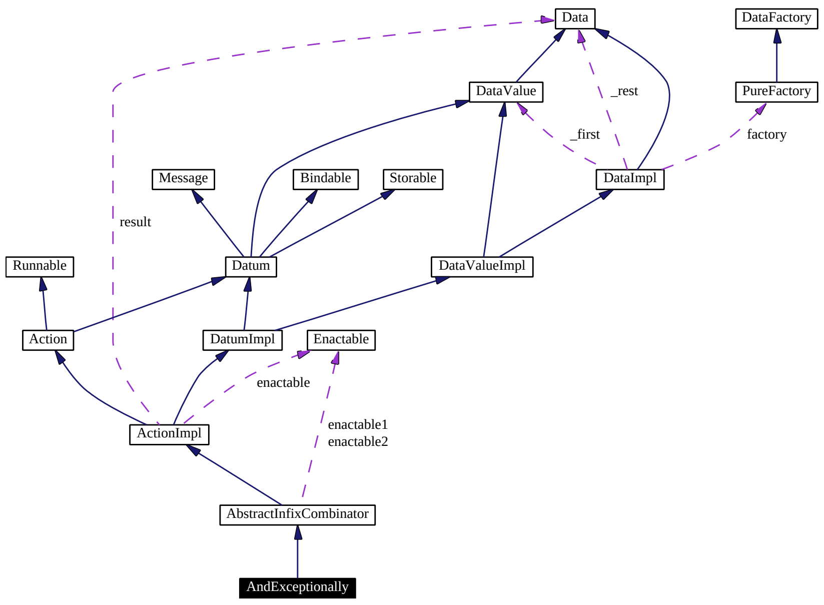
- AbstractAction.java
- ActionImpl.java

6.18 AndExceptionally Class Reference

Inheritance diagram for AndExceptionally:



Collaboration diagram for AndExceptionally:



Public Methods

- AndExceptionally (PureFactory factory, Enactable enactable1, Enactable enactable2)
- Data enact (Data data, Bindings bindings) throws Exceptional, Failed
- AndExceptionally (PureFactory factory, Enactable enactable1, Enactable enactable2)

6.18.1 Constructor & Destructor Documentation

6.18.1.1 AndExceptionally::AndExceptionally (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) [inline]

Definition at line 110 of file AbstractAction.java.

```
                                                                                                                                                        {
            super(factory, enactable1, enactable2);
        }
```

6.18.1.2 AndExceptionally::AndExceptionally (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) [inline]

Definition at line 154 of file ActionImpl.java.

```
                                                                                                                                                        {
            super(factory, enactable1, enactable2);
            enactable = new _AndExceptionally();
        }
```

6.18.2 Member Function Documentation

6.18.2.1 Data AndExceptionally::enact (Data *data*, Bindings *bindings*) [inline]

Definition at line 113 of file AbstractAction.java.

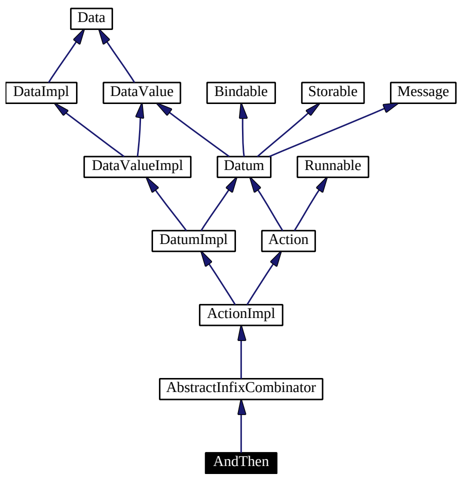
```
                                                                                                                                    {
            try {
                return enactable1.enact(data,bindings);
            }
            catch (Exceptional e1) {
                Data d1 = e1.getData();
                try {
                    return enactable2.enact(data, bindings);
                }
                catch (Exceptional e2) {
                    Data d2 = e2.getData();
                    throw new Exceptional(d1.merge(d2));
                }
            }
        }
```

The documentation for this class was generated from the following files:

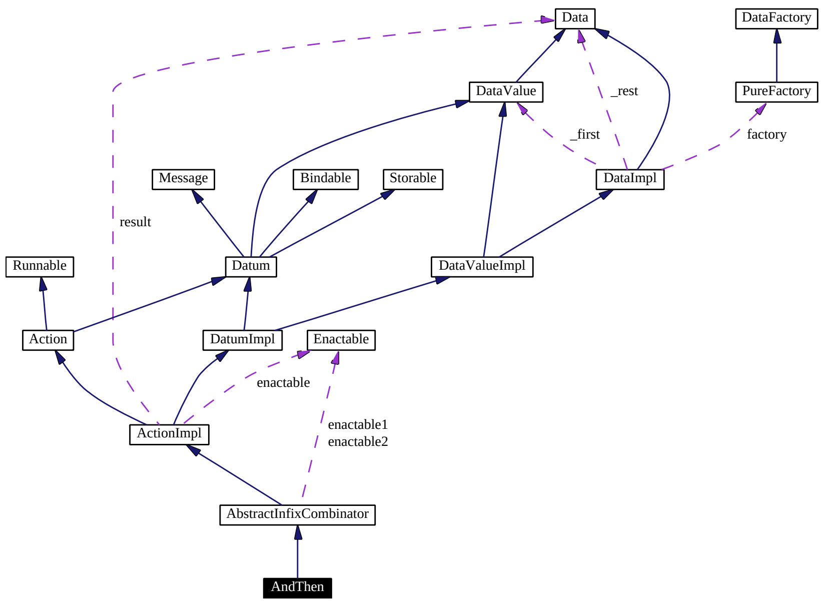
- AbstractAction.java
- ActionImpl.java

6.19 AndThen Class Reference

Inheritance diagram for AndThen:



Collaboration diagram for AndThen:



Public Methods

- AndThen (PureFactory factory, Enactable enactable1, Enactable enactable2)
- Data enact (Data data, Bindings bindings) throws Exceptional, Failed
- AndThen (PureFactory factory, Enactable enactable1, Enactable enactable2)

6.19.1 Constructor & Destructor Documentation

6.19.1.1 AndThen::AndThen (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 73 of file AbstractAction.java.

```
                                                                                {
        super(factory, enactable1, enactable2);
    }
```

6.19.1.2 AndThen::AndThen (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 98 of file ActionImpl.java.

```
                                                                                {
        super(factory, enactable1, enactable2);
        enactable = new _AndThen();
    }
```

6.19.2 Member Function Documentation

6.19.2.1 Data AndThen::enact (Data *data*, Bindings *bindings*) `[inline]`

Definition at line 76 of file AbstractAction.java.

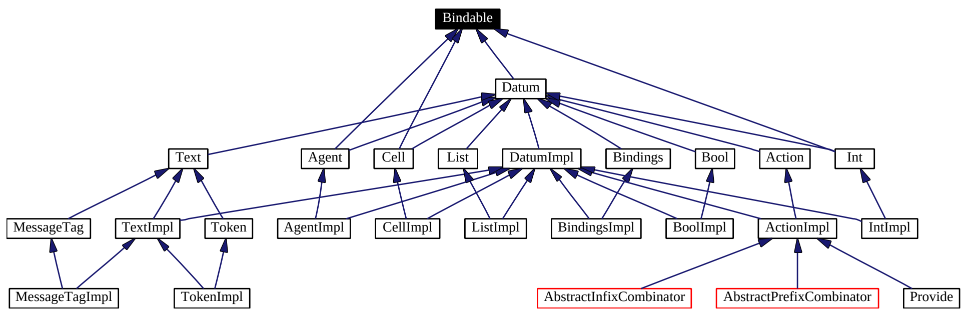
```
                                                                            {
        Data d1 = enactable1.enact(data, bindings);
        Data d2 = enactable2.enact(data, bindings);
        return d1.merge(d2);
    }
```

The documentation for this class was generated from the following files:

- AbstractAction.java
- ActionImpl.java

6.20 Bindable Interface Reference

Inheritance diagram for Bindable:

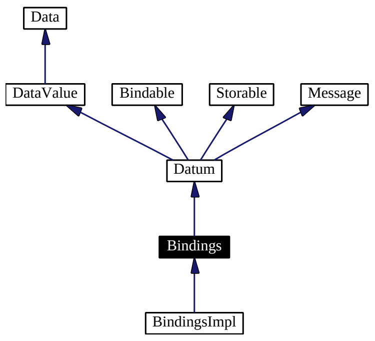


The documentation for this interface was generated from the following file:

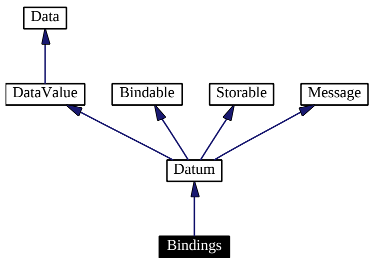
- Bindable.java

6.21 Bindings Interface Reference

Inheritance diagram for Bindings:



Collaboration diagram for Bindings:



Public Methods

- Map mapValue ()
- Bindings binding (Token token, Bindable bindable)
- Bindable bound (Token token) throws Exceptional
- Bindings overriding (Bindings bindings)
- Bindings disjointUnion (Bindings bindings)

6.21.1 Member Function Documentation

6.21.1.1 Bindings Bindings::binding (Token *token*, Bindable *bindable*)

Reimplemented in BindingsImpl.

6.21.1.2 Bindable Bindings::bound (Token *token*)

Reimplemented in BindingsImpl.

6.21.1.3 Bindings Bindings::disjointUnion (Bindings *bindings*)

Reimplemented in BindingsImpl.

6.21.1.4 Map Bindings::mapValue ()

Reimplemented in BindingsImpl.

Referenced by BindingsImpl::disjointUnion(), and BindingsImpl::overriding().

6.21.1.5 Bindings Bindings::overriding (Bindings *bindings*)

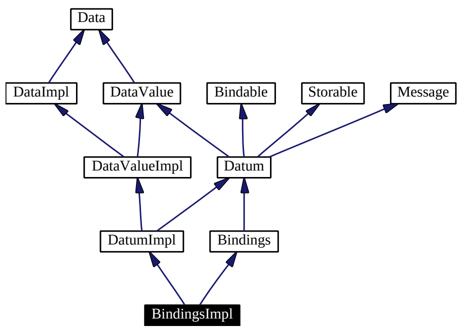
Reimplemented in BindingsImpl.

The documentation for this interface was generated from the following file:

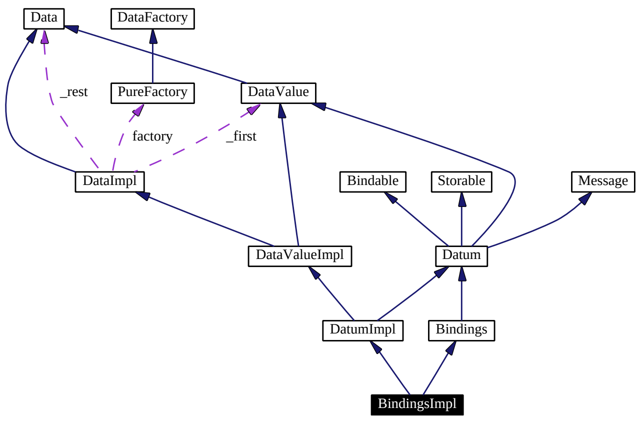
- Bindings.java

6.22 BindingsImpl Class Reference

Inheritance diagram for BindingsImpl:



Collaboration diagram for BindingsImpl:



Public Methods

- Map mapValue ()
- Bindings binding (Token token, Bindable bindable)
- Bindable bound (Token token) throws Exceptional
- Bindings overriding (Bindings bindings)
- Bindings disjointUnion (Bindings bindings)

Private Methods

- BindingsImpl (PureFactory factory)
- BindingsImpl (PureFactory factory, Map map)

Private Attributes

- Map bindingMap

6.22.1 Constructor & Destructor Documentation

6.22.1.1 BindingsImpl::BindingsImpl (PureFactory *factory*) `[inline, private]`

Definition at line 15 of file BindingsImpl.java.

```
00015                                       {
00016         this(factory, new TreeMap());
00017     }
```

6.22.1.2 BindingsImpl::BindingsImpl (PureFactory *factory*, Map *map*) `[inline, private]`

Definition at line 19 of file BindingsImpl.java.

```
00019                                                            {
00020         super(factory);
00021         bindingMap = map;
00022     }
```

6.22.2 Member Function Documentation

6.22.2.1 Bindings BindingsImpl::binding (Token *token*, Bindable *bindable*) `[inline]`

Reimplemented from Bindings.

Definition at line 28 of file BindingsImpl.java.

```
00028                                                                 {
00029         Map map = new TreeMap();
00030         map.put(token, bindable);
00031         return new BindingsImpl(factory, map);
00032     }
```

6.22.2.2 Bindable BindingsImpl::bound (Token *token*) `[inline]`

Reimplemented from Bindings.

Definition at line 34 of file BindingsImpl.java.

```
00034                                                               {
00035         if (bindingMap.containsKey(token))
00036             return (Bindable)bindingMap.get(token);
00037         throw new Exceptional(factory.makeEmpty());
00038     }
```

6.22.2.3 Bindings BindingsImpl::disjointUnion (Bindings *bindings*) `[inline]`

Reimplemented from Bindings.

Definition at line 47 of file BindingsImpl.java.

```
00047                                                           {
00048         Map newMap = new TreeMap(bindingMap);
00049         Set keySet = newMap.keySet();
00050         for (Iterator i = keySet.iterator(); i.hasNext();) {
00051             Token token = (Token)i.next();
00052             if (bindings.mapValue().containsKey(token))
00053                 newMap.remove(token);
00054             else
00055                 newMap.put(token, bindings.mapValue().get(token));
00056         }
00057         return new BindingsImpl(factory, newMap);
00058     }
```

6.22.2.4 Map BindingsImpl::mapValue () `[inline]`

Reimplemented from Bindings.

Definition at line 24 of file BindingsImpl.java.

```
00024                             {
00025         return bindingMap;
00026     }
```

6.22.2.5 Bindings BindingsImpl::overriding (Bindings *bindings*) `[inline]`

Reimplemented from Bindings.

Definition at line 41 of file BindingsImpl.java.

```
00041                                                     {
00042         Map newMap = new TreeMap(bindingMap);
00043         newMap.putAll(bindings.mapValue());
00044         return new BindingsImpl(factory, newMap);
00045     }
```

6.22.3 Member Data Documentation

6.22.3.1 Map BindingsImpl::bindingMap `[private]`

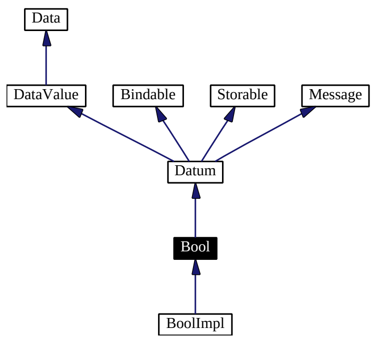
Definition at line 13 of file BindingsImpl.java.

The documentation for this class was generated from the following file:

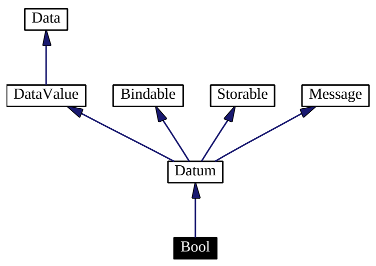
- BindingsImpl.java

6.23 Bool Interface Reference

Inheritance diagram for Bool:



Collaboration diagram for Bool:



Public Methods

- boolean booleanValue ()
- Bool not ()

6.23.1 Member Function Documentation

6.23.1.1 boolean Bool::booleanValue ()

Reimplemented in BoolImpl.

6.23.1.2 Bool Bool::not ()

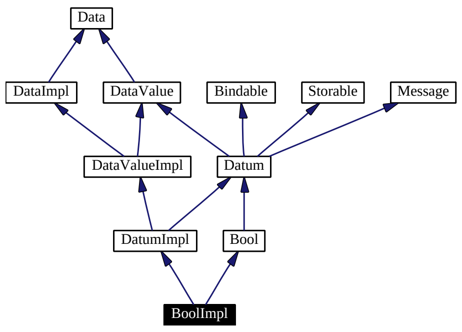
Reimplemented in BoolImpl.

The documentation for this interface was generated from the following file:

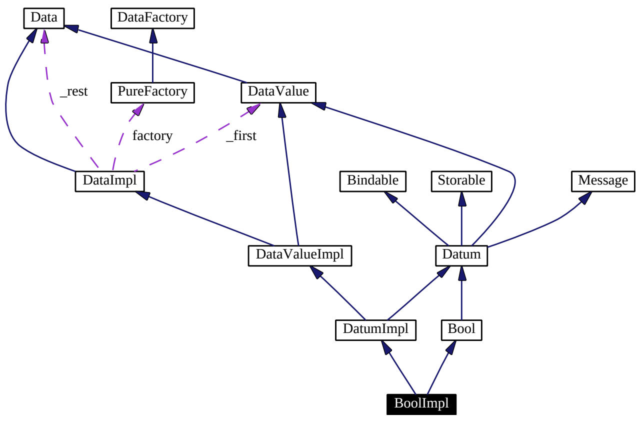
- Bool.java

6.24 BoolImpl Class Reference

Inheritance diagram for BoolImpl:



Collaboration diagram for BoolImpl:



Public Methods

- boolean booleanValue ()
- Bool not ()
- Empty equals (Data data) throws Exceptional

Private Methods

- BoolImpl (PureFactory factory, boolean value)

Private Attributes

- boolean value

6.24.1 Constructor & Destructor Documentation

6.24.1.1 BoolImpl::BoolImpl (PureFactory *factory*, boolean *value*) `[inline, private]`

Definition at line 12 of file BoolImpl.java.

```
00012                                                  {
00013         super(factory);
00014         this.value = value;
00015     }
```

6.24.2 Member Function Documentation

6.24.2.1 boolean BoolImpl::booleanValue () `[inline]`

Reimplemented from Bool.

Definition at line 17 of file BoolImpl.java.

```
00017                                   {
00018         return value;
00019     }
```

6.24.2.2 Empty BoolImpl::equals (Data *data*) `[inline]`

Reimplemented from DataValueImpl.

Definition at line 25 of file BoolImpl.java.

```
00025                                                       {
00026         if (data instanceof Bool)
00027             if (value == ((Bool)data).booleanValue())
00028                 return factory.makeEmpty();
00029         throw new Exceptional(factory.makeEmpty());
00030     }
```

6.24.2.3 Bool BoolImpl::not () `[inline]`

Reimplemented from Bool.

Definition at line 21 of file BoolImpl.java.

```
00021                       {
00022         return factory.makeBool(!value);
00023     }
```

6.24.3 Member Data Documentation

6.24.3.1 boolean BoolImpl::value `[private]`

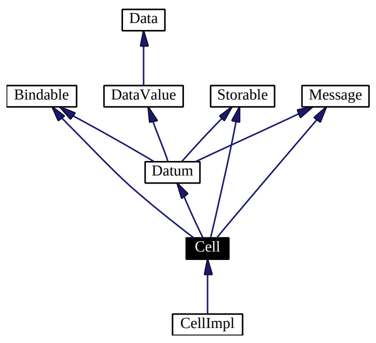
Definition at line 10 of file BoolImpl.java.

The documentation for this class was generated from the following file:

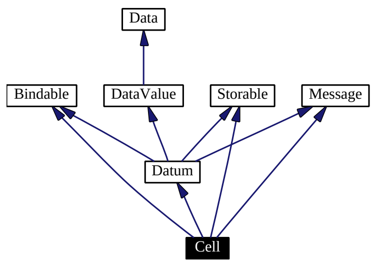
- BoolImpl.java

6.25 Cell Interface Reference

Inheritance diagram for Cell:



Collaboration diagram for Cell:

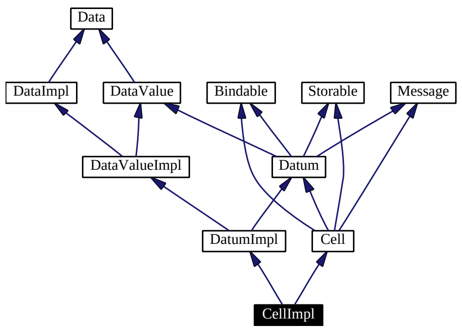


The documentation for this interface was generated from the following file:

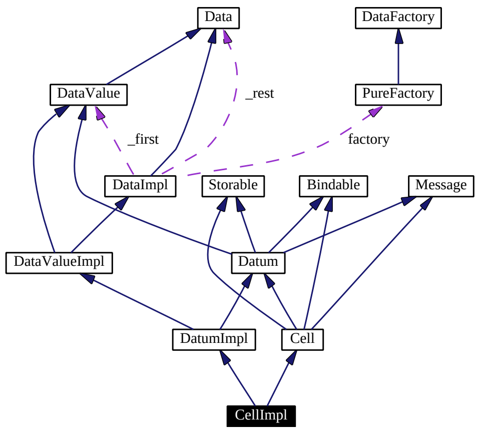
- Cell.java

6.26 CellImpl Class Reference

Inheritance diagram for CellImpl:



Collaboration diagram for CellImpl:



Private Methods

- CellImpl (PureFactory factory)

Private Attributes

- int id

Static Private Attributes

- int cells = 0

6.26.1 Constructor & Destructor Documentation

6.26.1.1 CellImpl::CellImpl (PureFactory *factory*) `[inline, private]`

Definition at line 11 of file CellImpl.java.

```
                                  {
         super(factory);
         id = cells++;
     }
```

6.26.2 Member Data Documentation

6.26.2.1 int CellImpl::cells = 0 `[static, private]`

Definition at line 8 of file CellImpl.java.

6.26.2.2 int CellImpl::id `[private]`

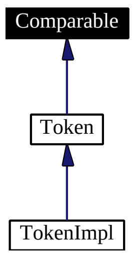
Definition at line 9 of file CellImpl.java.

The documentation for this class was generated from the following file:

- CellImpl.java

6.27 Comparable Class Reference

Inheritance diagram for Comparable:

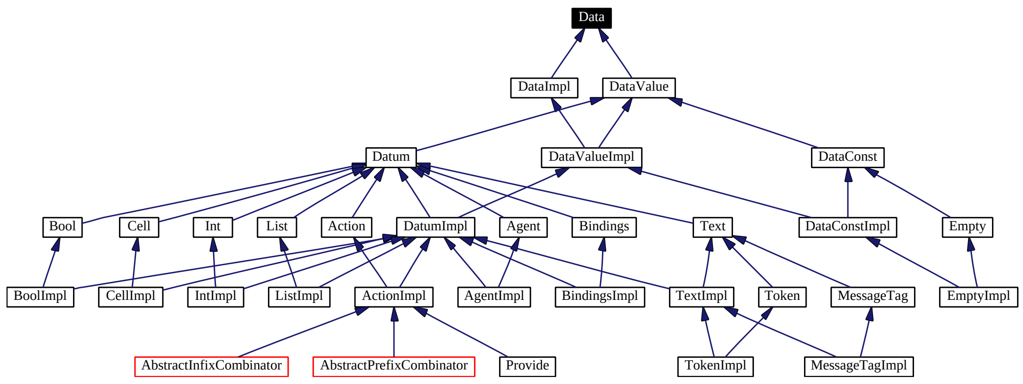


The documentation for this class was generated from the following file:

- Token.java

6.28 Data Interface Reference

Inheritance diagram for Data:



Public Methods

- Data concat (Data data)
- List tupleToList ()
- DataValue first () throws Exceptional
- Data rest ()
- DataValue component (int n) throws Exceptional
- Action provide ()

6.28.1 Member Function Documentation

6.28.1.1 DataValue Data::component (int *n*)

Reimplemented in DataImpl, and DataValueImpl.

Referenced by DataImpl::component().

6.28.1.2 Data Data::concat (Data *data*)

Reimplemented in DataImpl, and DataValueImpl.

Referenced by DataImpl::concat(), AndExceptionally::_AndExceptionally::enact(), And::_And::enact(), and AndThen::_AndThen::enact().

6.28.1.3 DataValue Data::first ()

Reimplemented in DataImpl, DataValueImpl, and EmptyImpl.

6.28.1.4 Action Data::provide ()

Reimplemented in DataImpl.

6.28.1.5 Data Data::rest ()

Reimplemented in DataImpl, and DataValueImpl.

6.28.1.6 List Data::tupleToList ()

Reimplemented in DataImpl, DataValueImpl, and EmptyImpl.

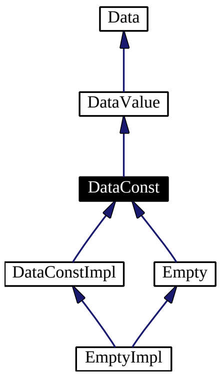
Referenced by DataImpl::tupleToList().

The documentation for this interface was generated from the following file:

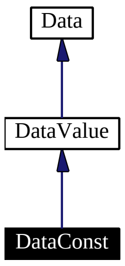
- Data.java

6.29 DataConst Interface Reference

Inheritance diagram for DataConst:



Collaboration diagram for DataConst:

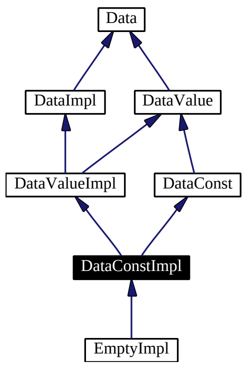


The documentation for this interface was generated from the following file:

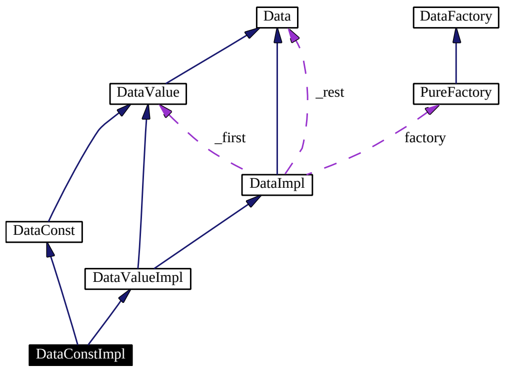
- DataConst.java

6.30 DataConstImpl Class Reference

Inheritance diagram for DataConstImpl:



Collaboration diagram for DataConstImpl:



Protected Methods

- DataConstImpl (PureFactory factory)

6.30.1 Constructor & Destructor Documentation

6.30.1.1 DataConstImpl::DataConstImpl (PureFactory *factory*) `[inline, protected]`

Definition at line 9 of file DataConstImpl.java.

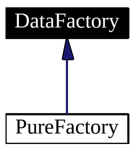
```
00009                                             {
00010         super(factory);
00011     }
```

The documentation for this class was generated from the following file:

- DataConstImpl.java

6.31 DataFactory Interface Reference

Inheritance diagram for DataFactory:



Public Methods

- Empty makeEmpty ()
- Data makeTuple (DataValue dataValue, Data data)
- Int makeInt (long n)
- Token makeToken (String s)
- Bool makeBool (boolean b)
- MessageTag makeMessageTag (String s)
- List makeList1 (DataValue dataValue)
- List makeList0 ()
- Action makeAction (Enactable enactable)
- Bindings makeNoBindings ()
- Agent makeAgent ()
- Cell makeCell ()

6.31.1 Member Function Documentation

6.31.1.1 Action DataFactory::makeAction (Enactable *enactable*)

Reimplemented in PureFactory.

6.31.1.2 Agent DataFactory::makeAgent ()

Reimplemented in PureFactory.

Referenced by Schedule::Schedule(), and Schedule::activate().

6.31.1.3 Bool DataFactory::makeBool (boolean *b*)

Reimplemented in PureFactory.

6.31.1.4 Cell DataFactory::makeCell ()

Reimplemented in PureFactory.

Referenced by Store::create().

6.31.1.5 Empty DataFactory::makeEmpty ()

Reimplemented in PureFactory.

Referenced by Schedule::deactivate(), TaggedBuffers::dequeue(), Store::destroy(), Store::inspect(), Schedule::receive(), Schedule::send(), and Store::update().

6.31.1.6 Int DataFactory::makeInt (long *n*)

Reimplemented in PureFactory.

Referenced by Schedule::chooseNatural(), and Schedule::giveCurrentTime().

6.31.1.7 List DataFactory::makeList0 ()

Reimplemented in PureFactory.

6.31.1.8 List DataFactory::makeList1 (DataValue *dataValue*)

Reimplemented in PureFactory.

6.31.1.9 MessageTag DataFactory::makeMessageTag (String *s*)

Reimplemented in PureFactory.

6.31.1.10 Bindings DataFactory::makeNoBindings ()

Reimplemented in PureFactory.

6.31.1.11 Token DataFactory::makeToken (String *s*)

Reimplemented in PureFactory.

6.31.1.12 Data DataFactory::makeTuple (DataValue *dataValue*, Data *data*)

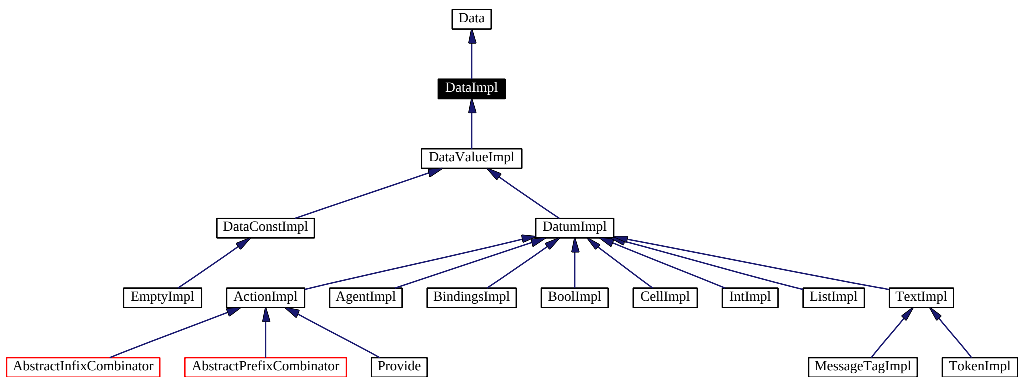
Reimplemented in PureFactory.

The documentation for this interface was generated from the following file:

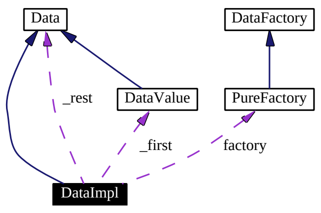
- DataFactory.java

6.32 DataImpl Class Reference

Inheritance diagram for DataImpl:



Collaboration diagram for DataImpl:



Public Methods

- Data concat (Data data)
- List tupleToList ()
- DataValue first () throws Exceptional
- Data rest ()
- DataValue component (int n) throws Exceptional
- Action provide ()

Protected Methods

- DataImpl (PureFactory factory, DataValue first, Data rest)
- DataImpl (PureFactory factory)

Protected Attributes

- DataValue _first
- Data _rest

Private Attributes

- PureFactory factory

6.32.1 Constructor & Destructor Documentation

6.32.1.1 DataImpl::DataImpl (PureFactory *factory*, DataValue *first*, Data *rest*) [inline, protected]

Definition at line 12 of file DataImpl.java.

```
00012                                                                          {
00013         this.factory = factory;
00014         this._first = first;
00015         this._rest = rest;
00016     }
```

6.32.1.2 DataImpl::DataImpl (PureFactory *factory*) [inline, protected]

Definition at line 18 of file DataImpl.java.

```
00018                                         {
00019         this(factory, null, null);
00020     }
```

6.32.2 Member Function Documentation

6.32.2.1 DataValue DataImpl::component (int *n*) [inline]

Reimplemented from Data.

Reimplemented in DataValueImpl.

Definition at line 40 of file DataImpl.java.

```
00040                                                               {
00041         if (n == 1)
00042             return _first;
00043         return _rest.component(n - 1);
00044     }
```

6.32.2.2 Data DataImpl::concat (Data *data*) [inline]

Reimplemented from Data.

Reimplemented in DataValueImpl.

Definition at line 24 of file DataImpl.java.

```
00024                                    {
00025         return new DataImpl(factory, _first, _rest.concat(data));
00026     }
```

6.32.2.3 DataValue DataImpl::first () [inline]

Reimplemented from Data.

Reimplemented in DataValueImpl, and EmptyImpl.

Definition at line 32 of file DataImpl.java.

```
00032                                                {
00033          return _first;
00034      }
```

6.32.2.4 Action DataImpl::provide () `[inline]`

Reimplemented from Data.

Definition at line 46 of file DataImpl.java.

```
00046                                  {
00047          return new Provide(factory, this);
00048      }
```

6.32.2.5 Data DataImpl::rest () `[inline]`

Reimplemented from Data.

Reimplemented in DataValueImpl.

Definition at line 36 of file DataImpl.java.

```
00036                             {
00037          return _rest;
00038      }
```

6.32.2.6 List DataImpl::tupleToList () `[inline]`

Reimplemented from Data.

Reimplemented in DataValueImpl, and EmptyImpl.

Definition at line 28 of file DataImpl.java.

```
00028                                    {
00029          return _rest.tupleToList().insert(_first);
00030      }
```

6.32.3 Member Data Documentation

6.32.3.1 DataValue DataImpl::_first `[protected]`

Definition at line 9 of file DataImpl.java.

6.32.3.2 Data DataImpl::_rest `[protected]`

Definition at line 10 of file DataImpl.java.

6.32.3.3 PureFactory DataImpl::factory `[private]`

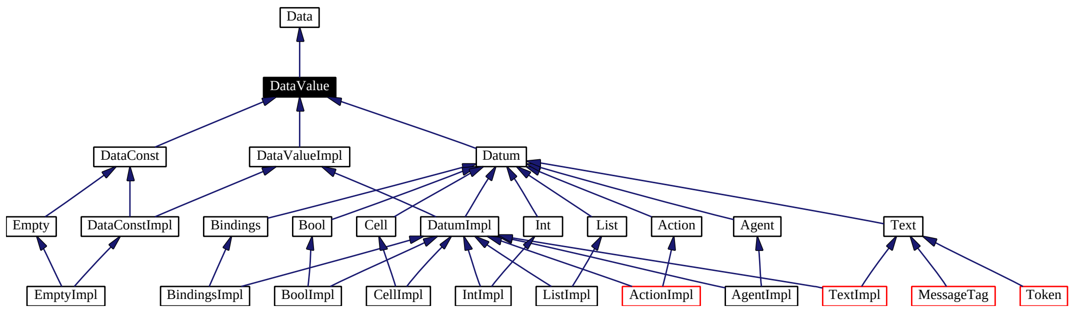
Definition at line 8 of file DataImpl.java.

The documentation for this class was generated from the following file:

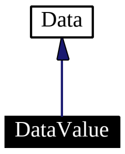
- DataImpl.java

6.33 DataValue Interface Reference

Inheritance diagram for DataValue:



Collaboration diagram for DataValue:



Public Methods

- Empty equals (Data data) throws Exceptional

6.33.1 Member Function Documentation

6.33.1.1 Empty DataValue::equals (Data *data*)

Reimplemented in ActionImpl, BoolImpl, DataValueImpl, IntImpl, ListImpl, and TextImpl.

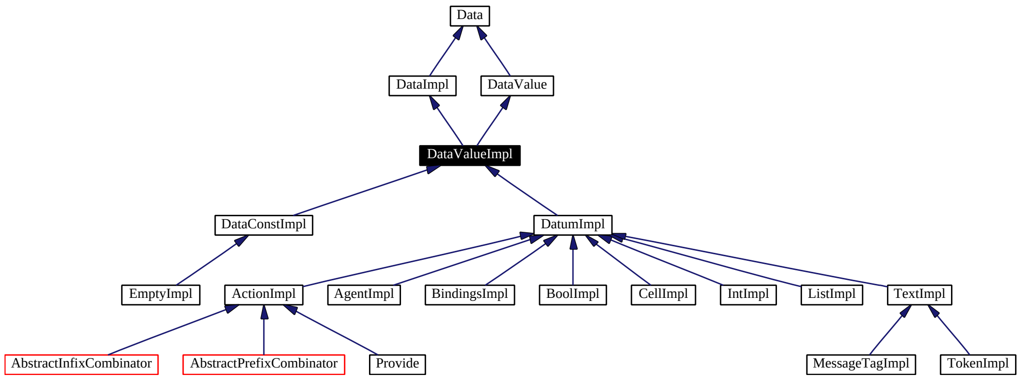
Referenced by ListImpl::equals().

The documentation for this interface was generated from the following file:

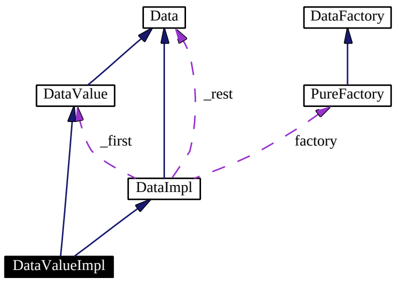
- DataValue.java

6.34 DataValueImpl Class Reference

Inheritance diagram for DataValueImpl:



Collaboration diagram for DataValueImpl:



Public Methods

- DataValue first () throws Exceptional
- Data rest ()
- List tupleToList ()
- Data concat (Data data)
- DataValue component (int n) throws Exceptional
- Empty equals (Data data) throws Exceptional

Protected Methods

- DataValueImpl (PureFactory factory)

6.34.1 Constructor & Destructor Documentation

6.34.1.1 DataValueImpl::DataValueImpl (PureFactory *factory*) `[inline, protected]`

Definition at line 9 of file DataValueImpl.java.

```
00009                                              {
00010         super(factory);
00011     }
```

6.34.2 Member Function Documentation

6.34.2.1 DataValue DataValueImpl::component (int *n*) `[inline]`

Reimplemented from DataImpl.

Definition at line 27 of file DataValueImpl.java.

```
00027                                                           {
00028         if (n == 1)
00029             return this;
00030         throw new Exceptional(factory.makeEmpty());
00031     }
```

6.34.2.2 Data DataValueImpl::concat (Data *data*) `[inline]`

Reimplemented from DataImpl.

Definition at line 23 of file DataValueImpl.java.

```
00023                                      {
00024         return factory.makeTuple(this, data);
00025     }
```

6.34.2.3 Empty DataValueImpl::equals (Data *data*) `[inline]`

Reimplemented from DataValue.

Reimplemented in ActionImpl, BoolImpl, IntImpl, ListImpl, and TextImpl.

Definition at line 37 of file DataValueImpl.java.

```
00037                                                        {
00038         if (this == data)
00039             return factory.makeEmpty();
00040         throw new Exceptional(factory.makeEmpty());
00041     }
```

6.34.2.4 DataValue DataValueImpl::first () `[inline]`

Reimplemented from DataImpl.

Reimplemented in EmptyImpl.

Definition at line 12 of file DataValueImpl.java.

```
00012                                                  {
00013         return this;
00014     }
```

6.34.2.5 Data DataValueImpl::rest () `[inline]`

Reimplemented from DataImpl.

Definition at line 15 of file DataValueImpl.java.

```
                        {
        return factory.makeEmpty();
    }
```

6.34.2.6 List DataValueImpl::tupleToList () [inline]

Reimplemented from DataImpl.

Reimplemented in EmptyImpl.

Definition at line 19 of file DataValueImpl.java.

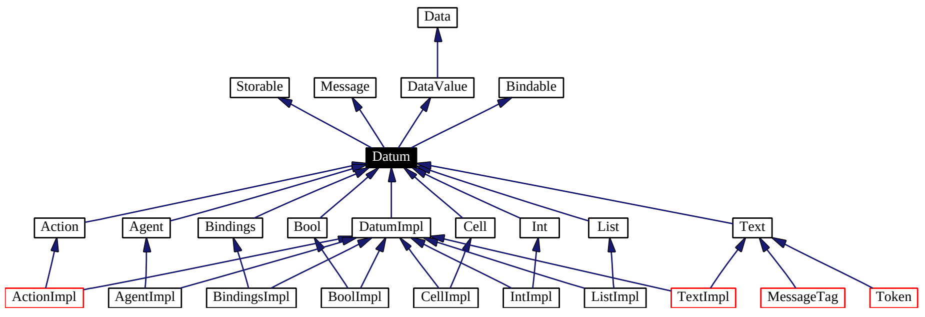
```
                               {
        return factory.makeList1(this);
    }
```

The documentation for this class was generated from the following file:

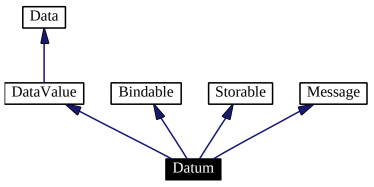
- DataValueImpl.java

6.35 Datum Interface Reference

Inheritance diagram for Datum:



Collaboration diagram for Datum:

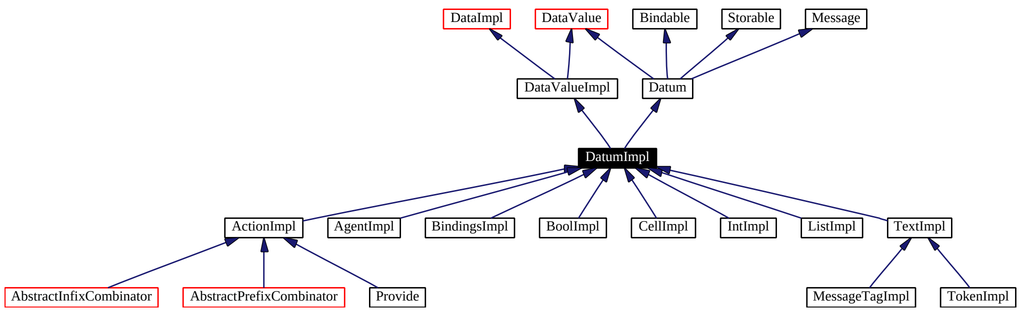


The documentation for this interface was generated from the following file:

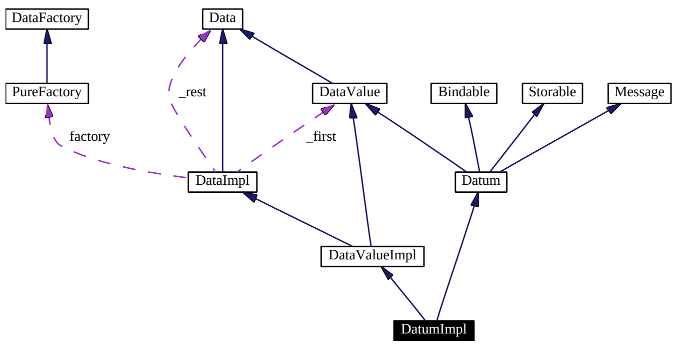
- Datum.java

6.36 DatumImpl Class Reference

Inheritance diagram for DatumImpl:



Collaboration diagram for DatumImpl:



Protected Methods

- DatumImpl (PureFactory factory)

6.36.1 Constructor & Destructor Documentation

6.36.1.1 DatumImpl::DatumImpl (PureFactory *factory*) `[inline, protected]`

Definition at line 8 of file DatumImpl.java.

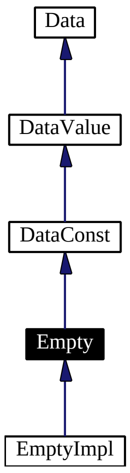
```
00008                                              {
00009          super(factory);
00010     }
```

The documentation for this class was generated from the following file:

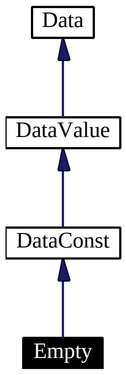
- DatumImpl.java

6.37 Empty Interface Reference

Inheritance diagram for Empty:



Collaboration diagram for Empty:

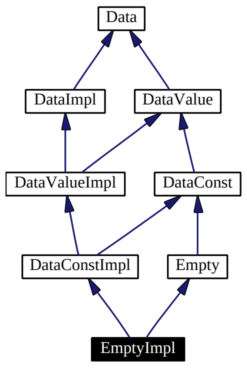


The documentation for this interface was generated from the following file:

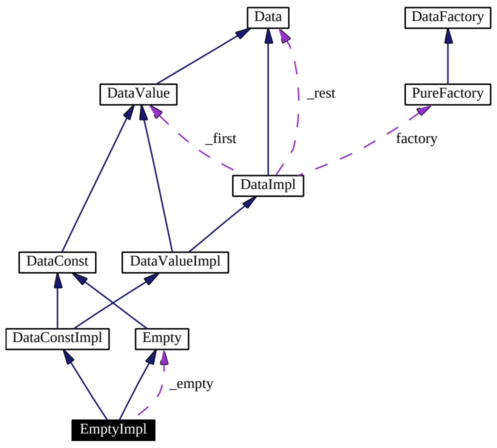
- Empty.java

6.38 EmptyImpl Class Reference

Inheritance diagram for EmptyImpl:



Collaboration diagram for EmptyImpl:



Public Methods

- DataValue first () throws Exceptional
- List tupleToList ()

Private Methods

- EmptyImpl (PureFactory factory)

Static Private Methods

- void init (PureFactory factory)
- Empty empty ()

Static Private Attributes

- Empty _empty = null

6.38.1 Constructor & Destructor Documentation

6.38.1.1 EmptyImpl::EmptyImpl (PureFactory *factory*) `[inline, private]`

Definition at line 11 of file EmptyImpl.java.

```
00011                                            {
00012           super(factory);
00013       }
```

6.38.2 Member Function Documentation

6.38.2.1 Empty EmptyImpl::empty () `[inline, static, private]`

Definition at line 20 of file EmptyImpl.java.

```
00020                             {
00021           return _empty;
00022       }
```

6.38.2.2 DataValue EmptyImpl::first () `[inline]`

Reimplemented from DataValueImpl.

Definition at line 26 of file EmptyImpl.java.

```
00026                                                  {
00027           throw new Exceptional(factory.makeEmpty());
00028       }
```

6.38.2.3 void EmptyImpl::init (PureFactory *factory*) `[inline, static, private]`

Definition at line 15 of file EmptyImpl.java.

```
00015                                           {
00016           _empty = new EmptyImpl(factory);
00017           //_empty._rest = _empty;
00018       }
```

6.38.2.4 List EmptyImpl::tupleToList () `[inline]`

Reimplemented from DataValueImpl.

Definition at line 30 of file EmptyImpl.java.

```
00030                                  {
00031           return factory.makeList0();
00032       }
```

6.38.3 Member Data Documentation

6.38.3.1 Empty EmptyImpl::_empty = null `[static, private]`

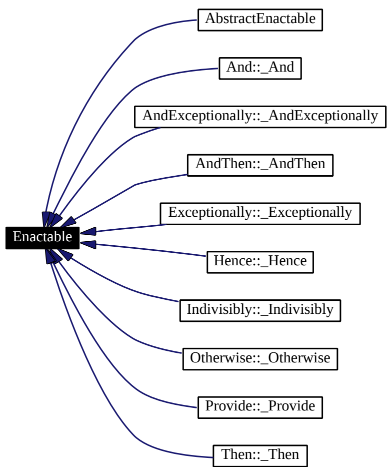
Definition at line 10 of file EmptyImpl.java.

The documentation for this class was generated from the following file:

- EmptyImpl.java

6.39 Enactable Interface Reference

Inheritance diagram for Enactable:



Public Methods

- Data enact (Data data, Bindings bindings) throws Exceptional, Failed

6.39.1 Member Function Documentation

6.39.1.1 Data Enactable::enact (Data *data*, Bindings *bindings*)

Reimplemented in Then::_Then, AndThen::_AndThen, And::_And, Exceptionally::_Exceptionally, AndExceptionally::_AndExceptionally, Otherwise::_Otherwise, Hence::_Hence, Indivisibly::_Indivisibly, and Provide::_Provide.

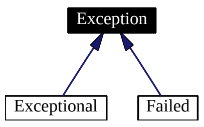
Referenced by Indivisibly::enact(), Hence::enact(), Otherwise::enact(), AndExceptionally::enact(), Exceptionally::enact(), And::enact(), AndThen::enact(), Then::enact(), and ActionImpl::enact().

The documentation for this interface was generated from the following file:

- Enactable.java

6.40 Exception Class Reference

Inheritance diagram for Exception:

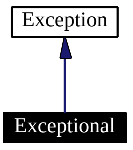


The documentation for this class was generated from the following file:

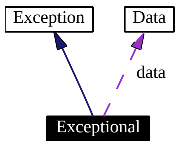
- Exceptional.java

6.41 Exceptional Class Reference

Inheritance diagram for Exceptional:



Collaboration diagram for Exceptional:



Public Methods

- Data getData ()
- Exceptional (Data data)
- Exceptional (String s, Data data)

Private Attributes

- Data data

6.41.1 Constructor & Destructor Documentation

6.41.1.1 Exceptional::Exceptional (Data *data*) `[inline]`

Definition at line 13 of file Exceptional.java.

```
00013                                    {
00014         super("Primitive action exception with data " + data);
00015         this.data = data;
00016     }
```

6.41.1.2 Exceptional::Exceptional (String *s*, Data *data*) `[inline]`

Definition at line 18 of file Exceptional.java.

```
00018                                              {
00019         super("Action exception: " + s + ", with data " + data);
00020         this.data = data;
00021     }
```

6.41.2 Member Function Documentation

6.41.2.1 Data Exceptional::getData () [inline]

Definition at line 9 of file Exceptional.java.

```
00009                             {
00010         return data;
00011     }
```

6.41.3 Member Data Documentation

6.41.3.1 Data Exceptional::data [private]

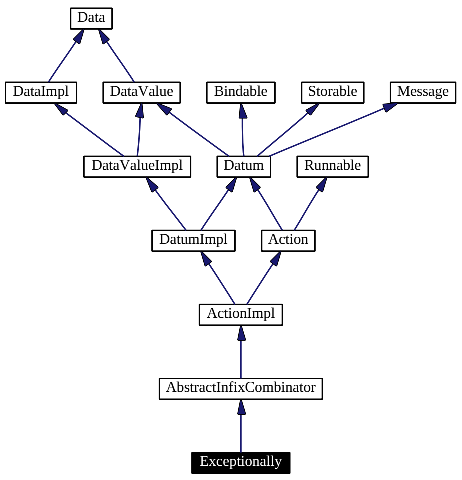
Definition at line 7 of file Exceptional.java.

The documentation for this class was generated from the following file:

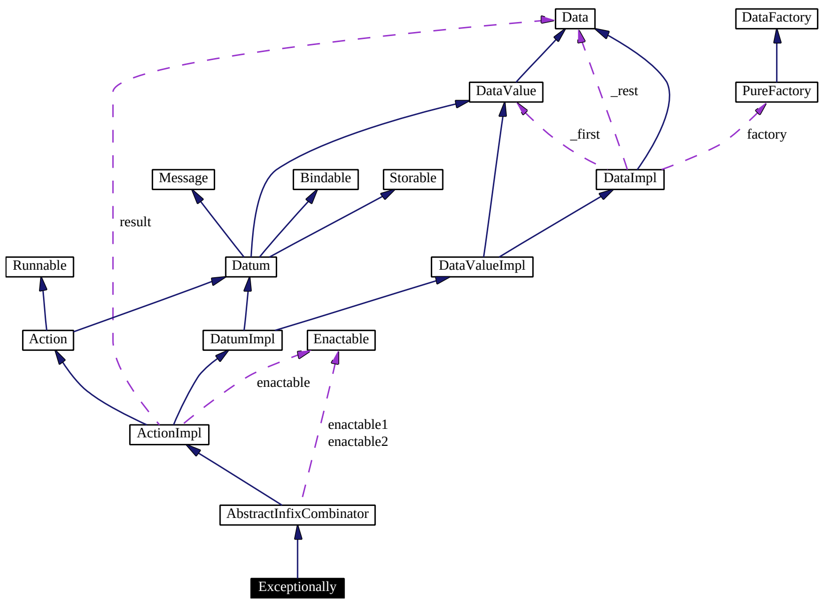
- Exceptional.java

6.42 Exceptionally Class Reference

Inheritance diagram for Exceptionally:



Collaboration diagram for Exceptionally:



Public Methods

- Exceptionally (PureFactory factory, Enactable enactable1, Enactable enactable2)
- Data enact (Data data, Bindings bindings) throws Exceptional, Failed
- Exceptionally (PureFactory factory, Enactable enactable1, Enactable enactable2)

6.42.1 Constructor & Destructor Documentation

6.42.1.1 Exceptionally::Exceptionally (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 95 of file AbstractAction.java.

```
                                                                        {
         super(factory, enactable1, enactable2);
     }
```

6.42.1.2 Exceptionally::Exceptionally (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 130 of file ActionImpl.java.

```
                                                                        {
         super(factory, enactable1, enactable2);
         enactable = new _Exceptionally();
     }
```

6.42.2 Member Function Documentation

6.42.2.1 Data Exceptionally::enact (Data *data*, Bindings *bindings*) `[inline]`

Definition at line 98 of file AbstractAction.java.

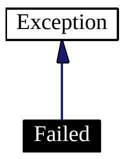
```
                                                                 {
         try {
             return enactable1.enact(data,bindings);
         }
         catch (Exceptional e) {
             Data d = e.getData();
             return enactable2.enact(d, bindings);
         }
     }
```

The documentation for this class was generated from the following files:

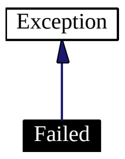
- AbstractAction.java
- ActionImpl.java

6.43 Failed Class Reference

Inheritance diagram for Failed:



Collaboration diagram for Failed:



Public Methods

- Failed ()

6.43.1 Constructor & Destructor Documentation

6.43.1.1 Failed::Failed () `[inline]`

Definition at line 5 of file Failed.java.

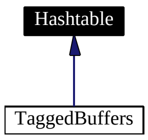
```
00005                    {
00006          super("Action failure");
00007     }
```

The documentation for this class was generated from the following file:

- Failed.java

6.44 Hashtable Class Reference

Inheritance diagram for Hashtable:

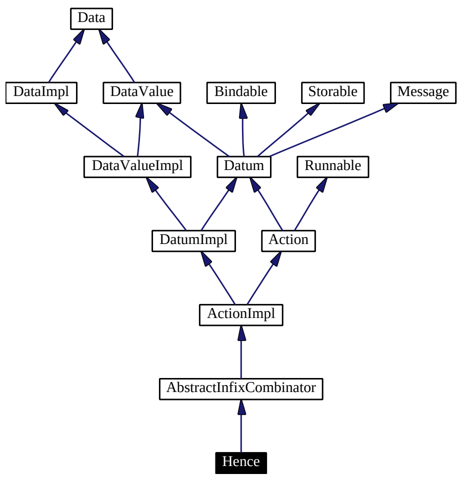


The documentation for this class was generated from the following file:

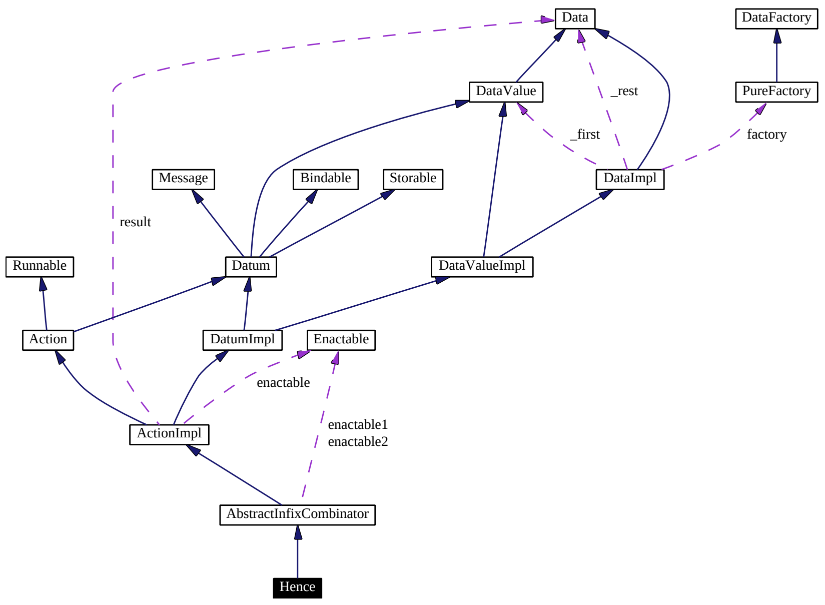
- TaggedBuffers.java

6.45 Hence Class Reference

Inheritance diagram for Hence:



Collaboration diagram for Hence:



Public Methods

- Hence (PureFactory factory, Enactable enactable1, Enactable enactable2)
- Data enact (Data data, Bindings bindings) throws Exceptional, Failed
- Hence (PureFactory factory, Enactable enactable1, Enactable enactable2)

6.45.1 Constructor & Destructor Documentation

6.45.1.1 Hence::Hence (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) [inline]

Definition at line 145 of file AbstractAction.java.

```
                                                                                {
        super(factory, enactable1, enactable2);
    }
```

6.45.1.2 Hence::Hence (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) [inline]

Definition at line 183 of file ActionImpl.java.

```
                                                                                {
        super(factory, enactable1, enactable2);
        enactable = new _Hence();
    }
```

6.45.2 Member Function Documentation

6.45.2.1 Data Hence::enact (Data *data*, Bindings *bindings*) [inline]

Definition at line 148 of file AbstractAction.java.

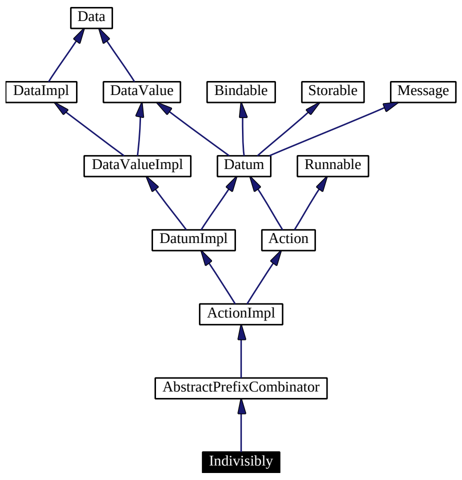
```
                                                                             {
        return enactable2.enact(data,
                                enactable1.enact(data, bindings).giveThe(Bindings));
    }
```

The documentation for this class was generated from the following files:

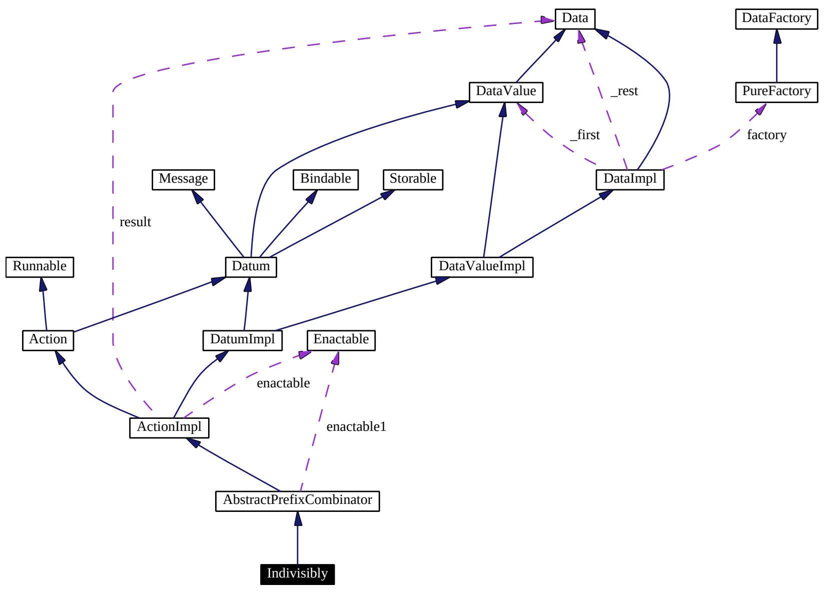
- AbstractAction.java
- ActionImpl.java

6.46 Indivisibly Class Reference

Inheritance diagram for Indivisibly:



Collaboration diagram for Indivisibly:



Public Methods

- Indivisibly (PureFactory factory, Enactable action)
- Data enact (Data data, Bindings bindings) throws Exceptional, Failed
- Indivisibly (PureFactory factory, Enactable enactable1)

6.46.1 Constructor & Destructor Documentation

6.46.1.1 Indivisibly::Indivisibly (PureFactory *factory*, Enactable *enactable1*) `[inline]`

Definition at line 156 of file AbstractAction.java.

```
                                                              {
        super(factory, action);
    }
```

6.46.1.2 Indivisibly::Indivisibly (PureFactory *factory*, Enactable *enactable1*) `[inline]`

Definition at line 198 of file ActionImpl.java.

```
                                                                  {
        super(factory, enactable1);
        enactable = new _Indivisibly();
    }
```

6.46.2 Member Function Documentation

6.46.2.1 Data Indivisibly::enact (Data *data*, Bindings *bindings*) `[inline]`

Definition at line 159 of file AbstractAction.java.

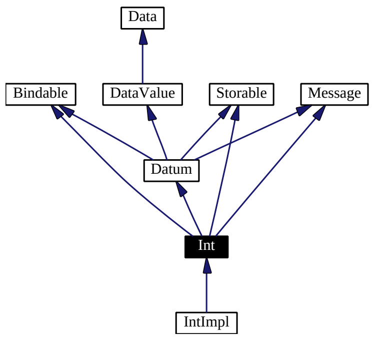
```
                                                                               {
        synchronized (Store.getStore()) {
            return action.enact(data, bindings);
        }
    }
```

The documentation for this class was generated from the following files:

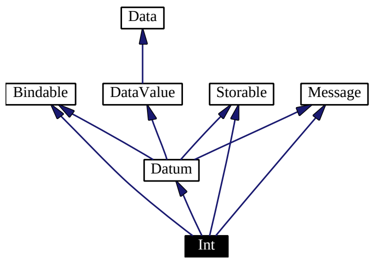
- AbstractAction.java
- ActionImpl.java

6.47 Int Interface Reference

Inheritance diagram for Int:



Collaboration diagram for Int:



Public Methods

- long intValue ()
- Int plus (Int n)
- Int minus (Int n)
- Int monus (Int n)
- Int times (Int n)
- Empty greater (Int n) throws Exceptional
- Empty greaterOrEq (Int n) throws Exceptional
- Empty less (Int n) throws Exceptional
- Empty lessOrEq (Int n) throws Exceptional

6.47.1 Member Function Documentation

6.47.1.1 Empty Int::greater (Int *n*)

Reimplemented in IntImpl.

6.47.1.2 Empty Int::greaterOrEq (Int *n*)

Reimplemented in IntImpl.

6.47.1.3 long Int::intValue ()

Reimplemented in IntImpl.

Referenced by IntImpl::greater(), IntImpl::greaterOrEq(), IntImpl::less(), IntImpl::lessOrEq(), IntImpl::minus(), IntImpl::monus(), IntImpl::plus(), and IntImpl::times().

6.47.1.4 Empty Int::less (Int *n*)

Reimplemented in IntImpl.

6.47.1.5 Empty Int::lessOrEq (Int *n*)

Reimplemented in IntImpl.

6.47.1.6 Int Int::minus (Int *n*)

Reimplemented in IntImpl.

6.47.1.7 Int Int::monus (Int *n*)

Reimplemented in IntImpl.

6.47.1.8 Int Int::plus (Int *n*)

Reimplemented in IntImpl.

6.47.1.9 Int Int::times (Int *n*)

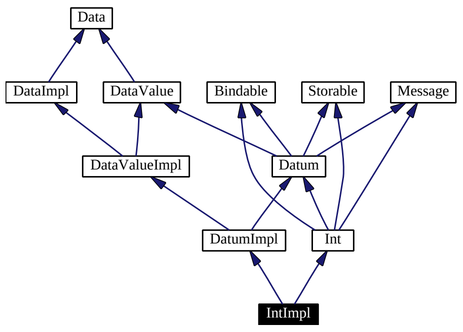
Reimplemented in IntImpl.

The documentation for this interface was generated from the following file:

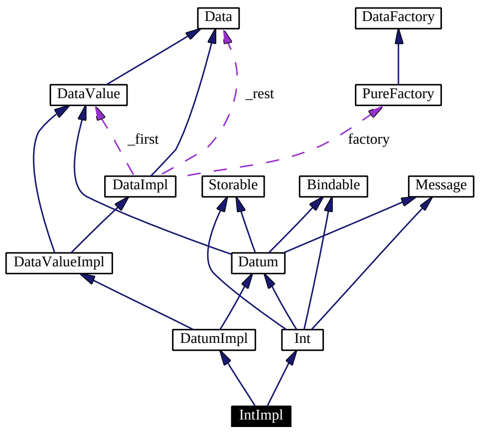
- Int.java

6.48 IntImpl Class Reference

Inheritance diagram for IntImpl:



Collaboration diagram for IntImpl:



Public Methods

- long intValue ()
- Int plus (Int n)
- Int minus (Int n)
- Int monus (Int n)
- Int times (Int n)
- Empty greater (Int n) throws Exceptional
- Empty greaterOrEq (Int n) throws Exceptional
- Empty less (Int n) throws Exceptional
- Empty lessOrEq (Int n) throws Exceptional
- Empty equals (Data data) throws Exceptional

Protected Methods

- IntImpl (PureFactory factory, long value)

Private Attributes

- long value

6.48.1 Constructor & Destructor Documentation

6.48.1.1 IntImpl::IntImpl (PureFactory *factory*, long *value*) `[inline, protected]`

Definition at line 10 of file IntImpl.java.

```
00010                                                         {
00011          super(factory);
00012          this.value = value;
00013      }
```

6.48.2 Member Function Documentation

6.48.2.1 Empty IntImpl::equals (Data *data*) `[inline]`

Reimplemented from DataValueImpl.

Definition at line 56 of file IntImpl.java.

```
00056                                                        {
00057          if (data instanceof Int)
00058              if (value == ((Int)data).intValue())
00059                  return factory.makeEmpty();
00060          throw new Exceptional(factory.makeEmpty());
00061      }
```

6.48.2.2 Empty IntImpl::greater (Int *n*) `[inline]`

Reimplemented from Int.

Definition at line 35 of file IntImpl.java.

```
00035                                                    {
00036          if (value > n.intValue())
00037              return factory.makeEmpty();
00038          throw new Exceptional(factory.makeEmpty());
00039      }
```

6.48.2.3 Empty IntImpl::greaterOrEq (Int *n*) `[inline]`

Reimplemented from Int.

Definition at line 40 of file IntImpl.java.

```
00040                                                         {
00041          if (value >= n.intValue())
00042              return factory.makeEmpty();
00043          throw new Exceptional(factory.makeEmpty());
00044      }
```

6.48.2.4 long IntImpl::intValue () `[inline]`

Reimplemented from Int.

Definition at line 16 of file IntImpl.java.

```
00016                            {
00017         return value;
00018     }
```

6.48.2.5 Empty IntImpl::less (Int *n*) `[inline]`

Reimplemented from Int.

Definition at line 45 of file IntImpl.java.

```
00045                                          {
00046         if (value < n.intValue())
00047             return factory.makeEmpty();
00048         throw new Exceptional(factory.makeEmpty());
00049     }
```

6.48.2.6 Empty IntImpl::lessOrEq (Int *n*) `[inline]`

Reimplemented from Int.

Definition at line 50 of file IntImpl.java.

```
00050                                              {
00051         if (value <= n.intValue())
00052             return factory.makeEmpty();
00053         throw new Exceptional(factory.makeEmpty());
00054     }
```

6.48.2.7 Int IntImpl::minus (Int *n*) `[inline]`

Reimplemented from Int.

Definition at line 23 of file IntImpl.java.

```
00023                             {
00024         return new IntImpl(factory, value - n.intValue());
00025     }
```

6.48.2.8 Int IntImpl::monus (Int *n*) `[inline]`

Reimplemented from Int.

Definition at line 26 of file IntImpl.java.

```
00026                             {
00027         if (n.intValue() < value)
00028             return new IntImpl(factory, value - n.intValue());
00029         return new IntImpl(factory, 0);
00030     }
```

6.48.2.9 Int IntImpl::plus (Int *n*) `[inline]`

Reimplemented from Int.

Definition at line 20 of file IntImpl.java.

```
                            {
        return new IntImpl(factory, value + n.intValue());
    }
```

6.48.2.10 Int IntImpl::times (Int *n*) `[inline]`

Reimplemented from Int.

Definition at line 31 of file IntImpl.java.

```
                             {
        return new IntImpl(factory, value * n.intValue());
    }
```

6.48.3 Member Data Documentation

6.48.3.1 long IntImpl::value `[private]`

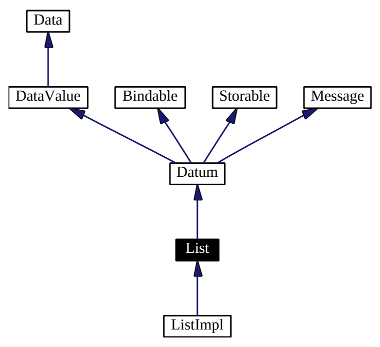
Definition at line 8 of file IntImpl.java.

The documentation for this class was generated from the following file:

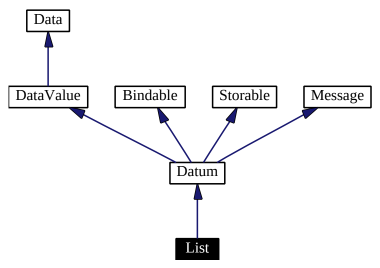
- IntImpl.java

6.49 List Interface Reference

Inheritance diagram for List:



Collaboration diagram for List:



Public Methods

- DataValue head () throws Exceptional
- List tail ()
- List insert (DataValue dataValue)
- List append (DataValue dataValue)
- List concat (List list)

6.49.1 Member Function Documentation

6.49.1.1 List List::append (DataValue *dataValue*)

Reimplemented in ListImpl.

Referenced by ListImpl::append().

6.49.1.2 List List::concat (List *list*)

Reimplemented in ListImpl.

Referenced by ListImpl::concat().

6.49.1.3 DataValue List::head ()

Reimplemented in ListImpl.

6.49.1.4 List List::insert (DataValue *dataValue*)

Reimplemented in ListImpl.

Referenced by DataImpl::tupleToList().

6.49.1.5 List List::tail ()

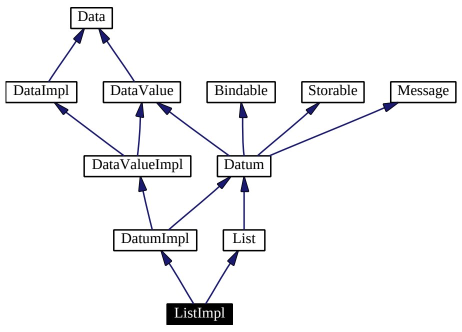
Reimplemented in ListImpl.

The documentation for this interface was generated from the following file:

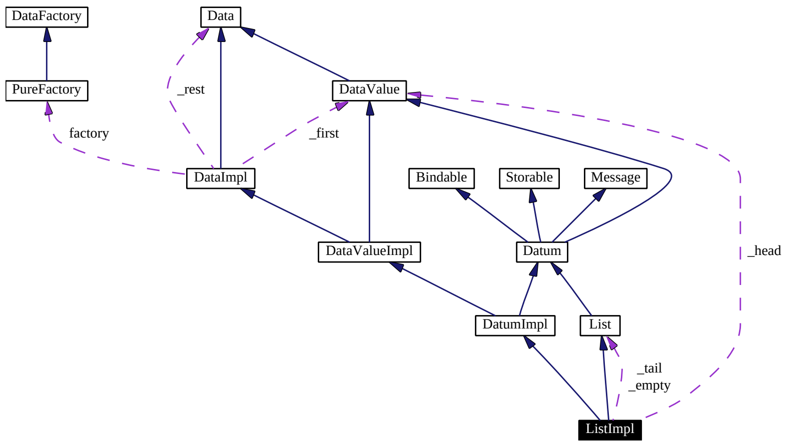
- List.java

6.50 ListImpl Class Reference

Inheritance diagram for ListImpl:



Collaboration diagram for ListImpl:



Public Methods

- DataValue head () throws Exceptional
- List tail ()
- List insert (DataValue dataValue)
- List append (DataValue dataValue)
- List concat (List list)
- Empty equals (Data data) throws Exceptional

Private Methods

- ListImpl (PureFactory factory)
- ListImpl (PureFactory factory, DataValue head)
- ListImpl (PureFactory factory, DataValue head, List tail)

Static Private Methods

- void init (PureFactory factory)
- List empty ()

Private Attributes

- DataValue _head
- List _tail

Static Private Attributes

- List _empty

6.50.1 Constructor & Destructor Documentation

6.50.1.1 ListImpl::ListImpl (PureFactory *factory*) `[inline, private]`

Definition at line 21 of file ListImpl.java.

```
00021                                           {
00022         this(factory, null, null);
00023     }
```

6.50.1.2 ListImpl::ListImpl (PureFactory *factory*, DataValue *head*) `[inline, private]`

Definition at line 25 of file ListImpl.java.

```
00025                                                   {
00026         this(factory, head, _empty);
00027     }
```

6.50.1.3 ListImpl::ListImpl (PureFactory *factory*, DataValue *head*, List *tail*) `[inline, private]`

Definition at line 29 of file ListImpl.java.

```
00029                                                              {
00030         super(factory);
00031         _head = head;
00032         _tail = tail;
00033     }
```

6.50.2 Member Function Documentation

6.50.2.1 List ListImpl::append (DataValue *dataValue*) `[inline]`

Reimplemented from List.

Definition at line 50 of file ListImpl.java.

```
00050                                             {
00051         if (this == _empty)
00052             return new ListImpl(factory, dataValue);
00053         return new ListImpl(factory, _head, _tail.append(dataValue));
00054     }
```

6.50.2.2 List ListImpl::concat (List *list*) [inline]

Reimplemented from List.

Definition at line 56 of file ListImpl.java.

```
00056                                   {
00057         if (list == _empty)
00058             return this;
00059         if (this == _empty)
00060             return list;
00061         return new ListImpl(factory, _head, _tail.concat(list));
00062     }
```

6.50.2.3 List ListImpl::empty () [inline, static, private]

Definition at line 17 of file ListImpl.java.

```
00017                         {
00018         return _empty;
00019     }
```

6.50.2.4 Empty ListImpl::equals (Data *data*) [inline]

Reimplemented from DataValueImpl.

Definition at line 64 of file ListImpl.java.

```
00064                                                       {
00065         if (this == data)
00066             return factory.makeEmpty();
00067         if (data instanceof List) {
00068             if (_head.equals(((List)data).head()) == factory.makeEmpty())
00069                 return _tail.equals(((List)data).tail());
00070         }
00071         throw new Exceptional(factory.makeEmpty());
00072     }
```

6.50.2.5 DataValue ListImpl::head () [inline]

Reimplemented from List.

Definition at line 36 of file ListImpl.java.

```
00036                                                {
00037         if (this == _empty)
00038             throw new Exceptional(factory.makeEmpty());
00039         return _head;
00040     }
```

6.50.2.6 void ListImpl::init (PureFactory *factory*) `[inline, static, private]`

Definition at line 13 of file ListImpl.java.

```
00013                                               {
00014         _empty = new ListImpl(factory);
00015     }
```

6.50.2.7 List ListImpl::insert (DataValue *dataValue*) `[inline]`

Reimplemented from List.

Definition at line 46 of file ListImpl.java.

```
00046                                                 {
00047         return new ListImpl(factory, dataValue, new ListImpl(factory, _head, _tail));
00048     }
```

6.50.2.8 List ListImpl::tail () `[inline]`

Reimplemented from List.

Definition at line 42 of file ListImpl.java.

```
00042                            {
00043         return _tail;
00044     }
```

6.50.3 Member Data Documentation

6.50.3.1 List ListImpl::_empty `[static, private]`

Definition at line 8 of file ListImpl.java.

6.50.3.2 DataValue ListImpl::_head `[private]`

Definition at line 10 of file ListImpl.java.

6.50.3.3 List ListImpl::_tail `[private]`

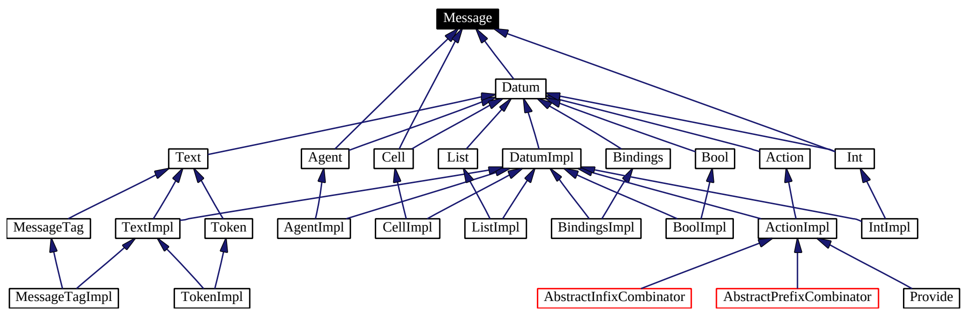
Definition at line 11 of file ListImpl.java.

The documentation for this class was generated from the following file:

- ListImpl.java

6.51 Message Interface Reference

Inheritance diagram for Message:

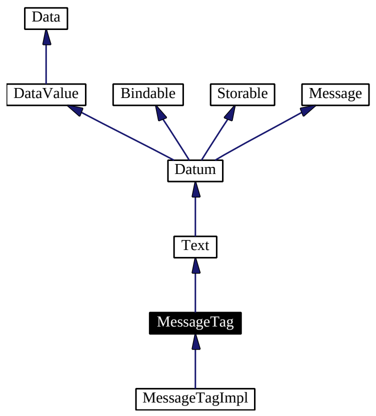


The documentation for this interface was generated from the following file:

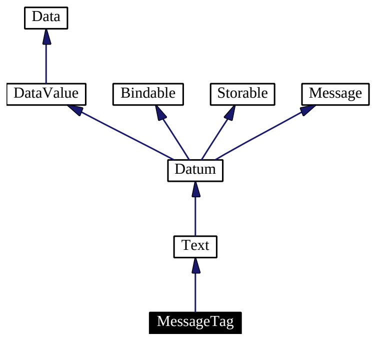
- Message.java

6.52 MessageTag Interface Reference

Inheritance diagram for MessageTag:



Collaboration diagram for MessageTag:

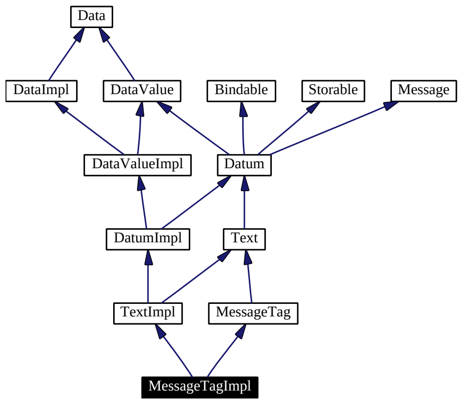


The documentation for this interface was generated from the following file:

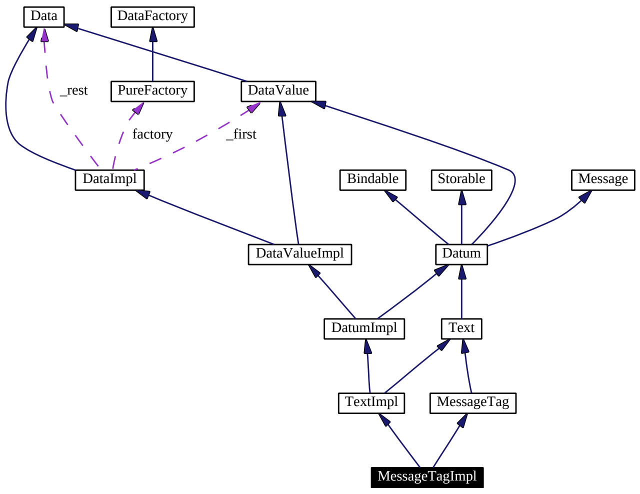
- MessageTag.java

6.53 MessageTagImpl Class Reference

Inheritance diagram for MessageTagImpl:



Collaboration diagram for MessageTagImpl:



Private Methods

- MessageTagImpl (PureFactory factory, String string)

6.53.1 Constructor & Destructor Documentation

6.53.1.1 MessageTagImpl::MessageTagImpl (PureFactory *factory*, String *string*) `[inline, private]`

Definition at line 11 of file MessageTagImpl.java.

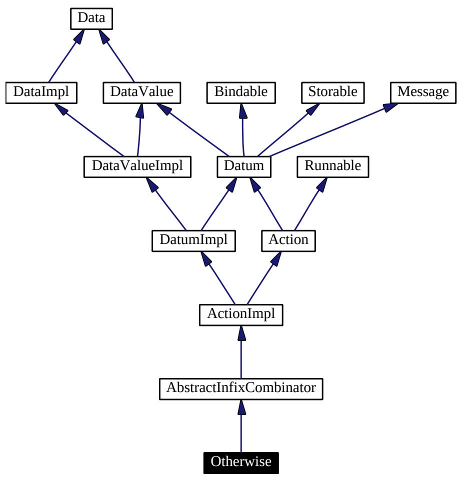
```
00011                                                         {
00012            super(factory, string);
00013       }
```

The documentation for this class was generated from the following file:

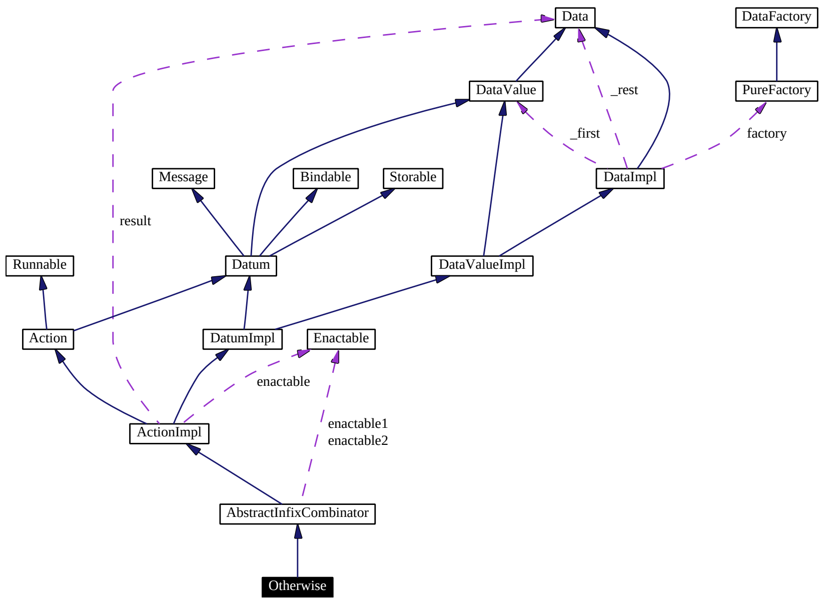
- MessageTagImpl.java

6.54 Otherwise Class Reference

Inheritance diagram for Otherwise:



Collaboration diagram for Otherwise:



Public Methods

- Otherwise (PureFactory factory, Enactable enactable1, Enactable enactable2)
- Data enact (Data data, Bindings bindings) throws Exceptional, Failed
- Otherwise (PureFactory factory, Enactable enactable1, Enactable enactable2)

6.54.1 Constructor & Destructor Documentation

6.54.1.1 Otherwise::Otherwise (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 131 of file AbstractAction.java.

```
00131                                                                                      {
00132            super(factory, enactable1, enactable2);
00133        }
```

6.54.1.2 Otherwise::Otherwise (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 171 of file ActionImpl.java.

```
00171                                                                                      {
00172            super(factory, enactable1, enactable2);
00173            enactable = new _Otherwise();
00174        }
```

6.54.2 Member Function Documentation

6.54.2.1 Data Otherwise::enact (Data *data*, Bindings *bindings*) `[inline]`

Definition at line 134 of file AbstractAction.java.

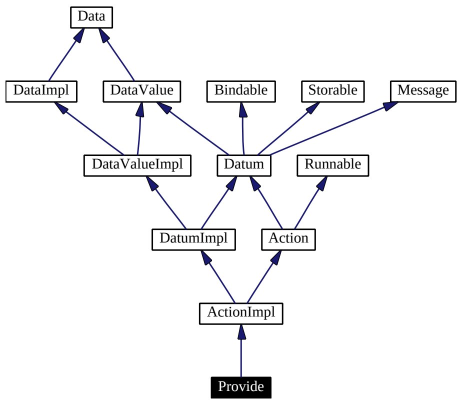
```
00134                                                                                {
00135            try {
00136                return enactable1.enact(data, bindings);
00137            }
00138            catch (Failed f) {
00139                return enactable2.enact(data, bindings);
00140            }
00141        }
```

The documentation for this class was generated from the following files:

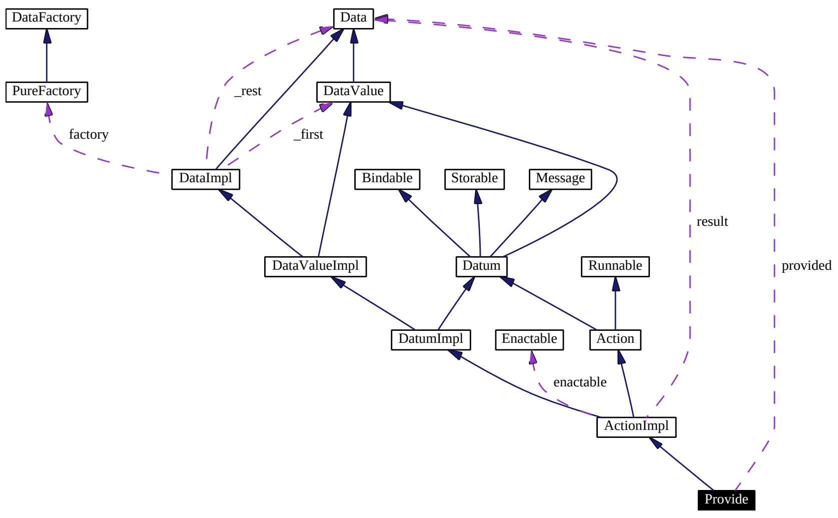
- AbstractAction.java
- ActionImpl.java

6.55 Provide Class Reference

Inheritance diagram for Provide:



Collaboration diagram for Provide:



Public Methods

- Provide (PureFactory factory, Data data)

Private Attributes

- Data provided

6.55.1 Constructor & Destructor Documentation

6.55.1.1 Provide::Provide (PureFactory *factory*, Data *data*) `[inline]`

Definition at line 59 of file DataImpl.java.

```
                                                    {
              super(factory);
              provided = data;
              enactable = new _Provide();
          }
```

6.55.2 Member Data Documentation

6.55.2.1 Data Provide::provided `[private]`

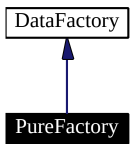
Definition at line 53 of file DataImpl.java.

The documentation for this class was generated from the following file:

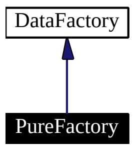
- DataImpl.java

6.56 PureFactory Class Reference

Inheritance diagram for PureFactory:



Collaboration diagram for PureFactory:



Public Methods

- PureFactory ()
- Int makeInt (long n)
- Token makeToken (String s)
- Bool makeBool (boolean b)
- MessageTag makeMessageTag (String s)
- List makeList0 ()
- List makeList1 (DataValue dataValue)
- Bindings makeNoBindings ()
- Agent makeAgent ()
- Cell makeCell ()
- Empty makeEmpty ()
- Data makeTuple (DataValue dataValue, Data data)
- Action makeAction (Enactable enactable)

6.56.1 Constructor & Destructor Documentation

6.56.1.1 PureFactory::PureFactory () `[inline]`

Definition at line 9 of file PureFactory.java.

```
00009                       {
00010         EmptyImpl.init(this);
00011         ListImpl.init(this);
00012     }
```

6.56.2 Member Function Documentation

6.56.2.1 Action PureFactory::makeAction (Enactable *enactable*) `[inline]`

Reimplemented from DataFactory.

Definition at line 51 of file PureFactory.java.

```
00051                                                  {
00052         return new ActionImpl(this, enactable);
00053     }
```

6.56.2.2 Agent PureFactory::makeAgent () [inline]

Reimplemented from DataFactory.

Definition at line 37 of file PureFactory.java.

```
00037                              {
00038         return new AgentImpl(this);
00039     }
```

6.56.2.3 Bool PureFactory::makeBool (boolean *b*) [inline]

Reimplemented from DataFactory.

Definition at line 20 of file PureFactory.java.

Referenced by BoolImpl::not().

```
00020                                     {
00021         return new BoolImpl(this, b);
00022     }
```

6.56.2.4 Cell PureFactory::makeCell () [inline]

Reimplemented from DataFactory.

Definition at line 40 of file PureFactory.java.

```
00040                            {
00041         return new CellImpl(this);
00042     }
```

6.56.2.5 Empty PureFactory::makeEmpty () [inline]

Reimplemented from DataFactory.

Definition at line 43 of file PureFactory.java.

Referenced by BindingsImpl::bound(), DataValueImpl::component(), ActionImpl::enact(), TextImpl::equals(), ListImpl::equals(), IntImpl::equals(), DataValueImpl::equals(), BoolImpl::equals(), ActionImpl::equals(), EmptyImpl::first(), IntImpl::greater(), IntImpl::greaterOrEq(), ListImpl::head(), IntImpl::less(), IntImpl::lessOrEq(), and DataValueImpl::rest().

```
00043                              {
00044         return EmptyImpl.empty();
00045     }
```

6.56.2.6 Int PureFactory::makeInt (long *n*) `[inline]`

Reimplemented from DataFactory.

Definition at line 14 of file PureFactory.java.

```
00014                                {
00015         return new IntImpl(this, n);
00016     }
```

6.56.2.7 List PureFactory::makeList0 () `[inline]`

Reimplemented from DataFactory.

Definition at line 26 of file PureFactory.java.

Referenced by EmptyImpl::tupleToList().

```
00026                             {
00027         return ListImpl.empty();
00028     }
```

6.56.2.8 List PureFactory::makeList1 (DataValue *dataValue*) `[inline]`

Reimplemented from DataFactory.

Definition at line 30 of file PureFactory.java.

Referenced by DataValueImpl::tupleToList().

```
00030                                                {
00031         return new ListImpl(this,dataValue);
00032     }
```

6.56.2.9 MessageTag PureFactory::makeMessageTag (String *s*) `[inline]`

Reimplemented from DataFactory.

Definition at line 23 of file PureFactory.java.

```
00023                                                {
00024         return new MessageTagImpl(this,s);
00025     }
```

6.56.2.10 Bindings PureFactory::makeNoBindings () `[inline]`

Reimplemented from DataFactory.

Definition at line 34 of file PureFactory.java.

Referenced by ActionImpl::enact().

```
00034                                      {
00035         return new BindingsImpl(this);
00036     }
```

6.56.2.11 Token PureFactory::makeToken (String *s*) [inline]

Reimplemented from DataFactory.

Definition at line 17 of file PureFactory.java.

```
                                      {
              return new TokenImpl(this, s);
          }
```

6.56.2.12 Data PureFactory::makeTuple (DataValue *dataValue*, Data *data*) [inline]

Reimplemented from DataFactory.

Definition at line 47 of file PureFactory.java.

Referenced by DataValueImpl::concat().

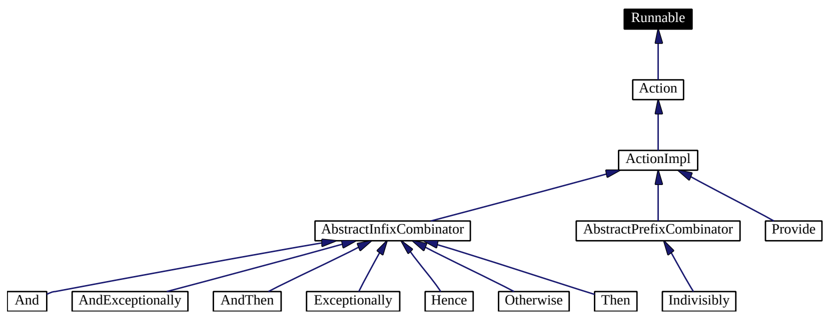
```
                                                                 {
              return new DataImpl(this, dataValue, data);
          }
```

The documentation for this class was generated from the following file:

- PureFactory.java

6.57 Runnable Class Reference

Inheritance diagram for Runnable:

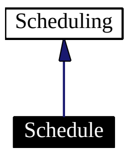


The documentation for this class was generated from the following file:

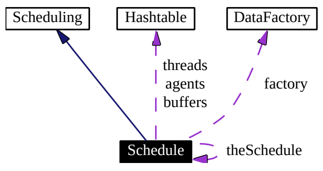
- Action.java

6.58 Schedule Class Reference

Inheritance diagram for Schedule:



Collaboration diagram for Schedule:



Public Methods

- Empty send (Agent agent, Message message, MessageTag messageTag) throws Exceptional
- Message receive (MessageTag messageTag) throws Exceptional
- Agent activate (Action action)
- Empty deactivate (Agent agent) throws Exceptional
- Agent giveCurrentAgent ()
- Int giveCurrentTime ()
- Int chooseNatural ()

Static Public Methods

- void initSchedule (DataFactory factory)
- Schedule getSchedule ()

Private Methods

- Schedule (DataFactory factory)

Private Attributes

- DataFactory factory
- Hashtable threads = new Hashtable()
- Hashtable agents = new Hashtable()
- Hashtable buffers = new Hashtable()

Static Private Attributes

- Schedule theSchedule = null

6.58.1 Constructor & Destructor Documentation

6.58.1.1 Schedule::Schedule (DataFactory *factory*) `[inline, private]`

Definition at line 19 of file Schedule.java.

```
                                           {
              this.factory = factory;
              Agent agent = factory.makeAgent();
              Thread thread = Thread.currentThread();
              agents.put(agent, thread);
              threads.put(thread, agent);
              buffers.put(agent, new TaggedBuffers(factory));
          }
```

6.58.2 Member Function Documentation

6.58.2.1 Agent Schedule::activate (Action *action*) `[inline]`

Reimplemented from Scheduling.

Definition at line 59 of file Schedule.java.

```
                                          {
              Agent agent = factory.makeAgent();
              Thread thread = new Thread(action);
              agents.put(agent, thread);
              threads.put(thread, agent);
              buffers.put(agent, new TaggedBuffers(factory));
              thread.start();
              return agent;
          }
```

6.58.2.2 Int Schedule::chooseNatural () `[inline]`

Reimplemented from Scheduling.

Definition at line 89 of file Schedule.java.

```
                               {
              return factory.makeInt((new Random()).nextInt());
          }
```

6.58.2.3 Empty Schedule::deactivate (Agent *agent*) `[inline]`

Reimplemented from Scheduling.

Definition at line 69 of file Schedule.java.

```
                                                              {
              if (agents.containsKey(agent)) {
                  Thread thread = (Thread)(agents.get(agent));
                  thread.interrupt();
                  agents.remove(agent);
                  threads.remove(thread);
                  return factory.makeEmpty();
              }
              throw new Exceptional(factory.makeEmpty());
          }
```

6.58.2.4 Schedule Schedule::getSchedule () [inline, static]

Definition at line 32 of file Schedule.java.

```
00032                                           {
00033         if (theSchedule == null)
00034             throw new RuntimeException("Initialize schedule first!");
00035         return theSchedule;
00036     }
```

6.58.2.5 Agent Schedule::giveCurrentAgent () [inline]

Reimplemented from Scheduling.

Definition at line 80 of file Schedule.java.

```
00080                                      {
00081         return (Agent)(threads.get(Thread.currentThread()));
00082     }
```

6.58.2.6 Int Schedule::giveCurrentTime () [inline]

Reimplemented from Scheduling.

Definition at line 84 of file Schedule.java.

```
00084                                   {
00085         return factory.makeInt((Calendar.getInstance()).get(Calendar.SECOND));
00086 
00087     }
```

6.58.2.7 void Schedule::initSchedule (DataFactory *factory*) [inline, static]

Definition at line 28 of file Schedule.java.

```
00028                                                           {
00029         theSchedule = new Schedule(factory);
00030     }
```

6.58.2.8 Message Schedule::receive (MessageTag *messageTag*) [inline]

Reimplemented from Scheduling.

Definition at line 48 of file Schedule.java.

```
00048                                                                       {
00049         Thread thread = Thread.currentThread();
00050         if (threads.containsKey(thread)) {
00051             Agent agent = (Agent)threads.get(thread);
00052             TaggedBuffers taggedBuffers = (TaggedBuffers)buffers.get(agent);
00053             return taggedBuffers.dequeue(messageTag);
00054         }
00055         throw new Exceptional(factory.makeEmpty());
00056     }
```

6.58.2.9 Empty Schedule::send (Agent *agent*, Message *message*, MessageTag *messageTag*) `[inline]`

Reimplemented from Scheduling.

Definition at line 40 of file Schedule.java.

```
00040                                                                                          {
00041           if (agents.containsKey(agent)) {
00042               TaggedBuffers taggedBuffers = (TaggedBuffers)buffers.get(agent);
00043               taggedBuffers.queue(messageTag, message);
00044           }
00045           throw new Exceptional(factory.makeEmpty());
00046      }
```

6.58.3 Member Data Documentation

6.58.3.1 Hashtable Schedule::agents = new Hashtable() `[private]`

Definition at line 16 of file Schedule.java.

6.58.3.2 Hashtable Schedule::buffers = new Hashtable() `[private]`

Definition at line 17 of file Schedule.java.

6.58.3.3 DataFactory Schedule::factory `[private]`

Definition at line 14 of file Schedule.java.

6.58.3.4 Schedule Schedule::theSchedule = null `[static, private]`

Definition at line 12 of file Schedule.java.

6.58.3.5 Hashtable Schedule::threads = new Hashtable() `[private]`

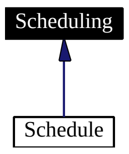
Definition at line 15 of file Schedule.java.

The documentation for this class was generated from the following file:

- Schedule.java

6.59 Scheduling Interface Reference

Inheritance diagram for Scheduling:



Public Methods

- Empty send (Agent agent, Message message, MessageTag messageTag) throws Exceptional
- Message receive (MessageTag messageTag) throws Exceptional
- Agent activate (Action action)
- Empty deactivate (Agent agent) throws Exceptional
- Agent giveCurrentAgent ()
- Int giveCurrentTime ()
- Int chooseNatural ()

6.59.1 Member Function Documentation

6.59.1.1 Agent Scheduling::activate (Action *action*)

Reimplemented in Schedule.

6.59.1.2 Int Scheduling::chooseNatural ()

Reimplemented in Schedule.

6.59.1.3 Empty Scheduling::deactivate (Agent *agent*)

Reimplemented in Schedule.

6.59.1.4 Agent Scheduling::giveCurrentAgent ()

Reimplemented in Schedule.

6.59.1.5 Int Scheduling::giveCurrentTime ()

Reimplemented in Schedule.

6.59.1.6 Message Scheduling::receive (MessageTag *messageTag*)

Reimplemented in Schedule.

6.59.1.7 Empty Scheduling::send (Agent *agent*, Message *message*, MessageTag *messageTag*)

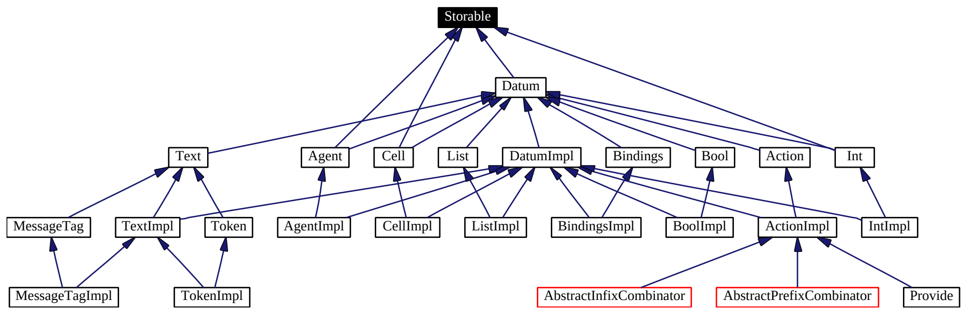
Reimplemented in Schedule.

The documentation for this interface was generated from the following file:

- Scheduling.java

6.60 Storable Interface Reference

Inheritance diagram for Storable:

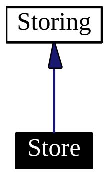


The documentation for this interface was generated from the following file:

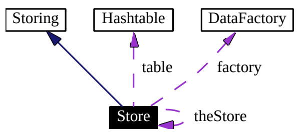
- Storable.java

6.61 Store Class Reference

Inheritance diagram for Store:



Collaboration diagram for Store:



Public Methods

- synchronized Cell create (Storable o)
- synchronized Empty destroy (Cell cell) throws Exceptional
- synchronized Empty update (Cell cell, Storable o) throws Exceptional
- synchronized Storable inspect (Cell cell) throws Exceptional
- String toString ()

Static Public Methods

- void initStore (DataFactory factory)
- void initStore (DataFactory factory, int cap)
- void initStore (DataFactory factory, int cap, int loadf)
- Store getStore ()

Private Methods

- Store (DataFactory factory, int cap, int loadf)

Private Attributes

- DataFactory factory
- Hashtable table

Static Private Attributes

- Store theStore = null

6.61.1 Constructor & Destructor Documentation

6.61.1.1 Store::Store (DataFactory *factory*, int *cap*, int *loadf*) `[inline, private]`

Definition at line 13 of file Store.java.

```
00013                                                               {
00014           this.factory = factory;
00015           table = new Hashtable(cap, loadf);
00016       }
```

6.61.2 Member Function Documentation

6.61.2.1 synchronized Cell Store::create (Storable *o*) `[inline]`

Reimplemented from Storing.

Definition at line 37 of file Store.java.

```
00037                                                    {
00038           Cell cell = factory.makeCell();
00039           table.put(cell, o);
00040           return cell;
00041       }
```

6.61.2.2 synchronized Empty Store::destroy (Cell *cell*) `[inline]`

Reimplemented from Storing.

Definition at line 43 of file Store.java.

```
00043                                                                        {
00044           if (table.remove(cell) == null)
00045               throw new Exceptional("destroy: cell non-existent", factory.makeEmpty());
00046           return factory.makeEmpty();
00047       }
```

6.61.2.3 Store Store::getStore () `[inline, static]`

Definition at line 30 of file Store.java.

```
00030                                      {
00031           if (theStore == null)
00032               throw new RuntimeException("Initialize store first!");
00033           return theStore;
00034       }
```

6.61.2.4 void Store::initStore (DataFactory *factory*, int *cap*, int *loadf*) `[inline, static]`

Definition at line 26 of file Store.java.

```
00026                                                                              {
00027           theStore = new Store(factory, cap, loadf);
00028       }
```

6.61.2.5 void Store::initStore (DataFactory *factory*, int *cap*) `[inline, static]`

Definition at line 22 of file Store.java.

```
00022                                                                 {
00023         initStore(factory, cap, 75);
00024     }
```

6.61.2.6 void Store::initStore (DataFactory *factory*) `[inline, static]`

Definition at line 18 of file Store.java.

```
00018                                                         {
00019         initStore(factory, 1024);
00020     }
```

6.61.2.7 synchronized Storable Store::inspect (Cell *cell*) `[inline]`

Reimplemented from Storing.

Definition at line 56 of file Store.java.

```
00056                                                                         {
00057         if (!table.containsKey(cell))
00058             throw new Exceptional("inpect: cell non-existent", factory.makeEmpty());
00059         return (Storable)(table.get(cell));
00060     }
```

6.61.2.8 String Store::toString () `[inline]`

Definition at line 62 of file Store.java.

```
00062                               {
00063         return table.toString();
00064     }
```

6.61.2.9 synchronized Empty Store::update (Cell *cell*, Storable *o*) `[inline]`

Reimplemented from Storing.

Definition at line 49 of file Store.java.

```
00049                                                                                 {
00050         if (!table.containsKey(cell))
00051             throw new Exceptional("update: cell non-existent", factory.makeEmpty());
00052         table.put(cell, o);
00053         return factory.makeEmpty();
00054     }
```

6.61.3 Member Data Documentation

6.61.3.1 DataFactory Store::factory `[private]`

Definition at line 10 of file Store.java.

6.61.3.2 Hashtable Store::table `[private]`

Definition at line 11 of file Store.java.

6.61.3.3 Store Store::theStore = null `[static, private]`

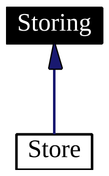
Definition at line 8 of file Store.java.

The documentation for this class was generated from the following file:

- Store.java

6.62 Storing Interface Reference

Inheritance diagram for Storing:



Public Methods

- Cell create (Storable storable)
- Empty destroy (Cell cell) throws Exceptional
- Empty update (Cell cell, Storable o) throws Exceptional
- Storable inspect (Cell cell) throws Exceptional

6.62.1 Member Function Documentation

6.62.1.1 Cell Storing::create (Storable *o*)

Reimplemented in Store.

6.62.1.2 Empty Storing::destroy (Cell *cell*)

Reimplemented in Store.

6.62.1.3 Storable Storing::inspect (Cell *cell*)

Reimplemented in Store.

6.62.1.4 Empty Storing::update (Cell *cell*, Storable *o*)

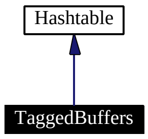
Reimplemented in Store.

The documentation for this interface was generated from the following file:

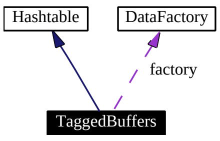
- Storing.java

6.63 TaggedBuffers Class Reference

Inheritance diagram for TaggedBuffers:



Collaboration diagram for TaggedBuffers:



Public Methods

- TaggedBuffers (DataFactory factory)
- void queue (MessageTag messageTag, Message message)
- Message dequeue (MessageTag messageTag) throws Exceptional

Private Attributes

- DataFactory factory

6.63.1 Constructor & Destructor Documentation

6.63.1.1 TaggedBuffers::TaggedBuffers (DataFactory *factory*) `[inline]`

Definition at line 11 of file TaggedBuffers.java.

```
                                               {
        super();
        this.factory = factory;
    }
```

6.63.2 Member Function Documentation

6.63.2.1 Message TaggedBuffers::dequeue (MessageTag *messageTag*) `[inline]`

Definition at line 26 of file TaggedBuffers.java.

Referenced by Schedule::receive().

```
                                                                       {
        if (!containsKey(messageTag))
            throw new Exceptional(factory.makeEmpty());
        return (Message)((LinkedList)get(messageTag)).removeFirst();
    }
```

6.63.2.2 void TaggedBuffers::queue (MessageTag *messageTag*, Message *message*) [inline]

Definition at line 16 of file TaggedBuffers.java.

Referenced by Schedule::send().

```
                                                                {
           if (!containsKey(messageTag)) {
               LinkedList messages = new LinkedList();
               messages.add(message);
               put(messageTag, messages);
           }
           else
               ((LinkedList)get(messageTag)).add(message);
       }
```

6.63.3 Member Data Documentation

6.63.3.1 DataFactory TaggedBuffers::factory [private]

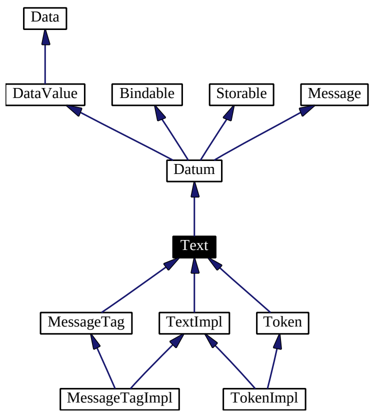
Definition at line 10 of file TaggedBuffers.java.

The documentation for this class was generated from the following file:

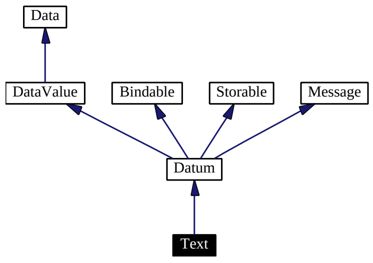
- TaggedBuffers.java

6.64 Text Interface Reference

Inheritance diagram for Text:



Collaboration diagram for Text:



Public Methods

- String stringValue ()

6.64.1 Member Function Documentation

6.64.1.1 String Text::stringValue ()

Reimplemented in TextImpl.

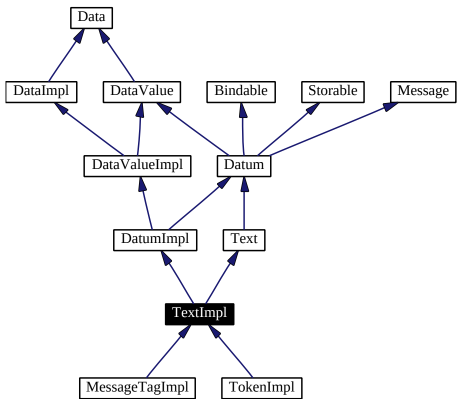
Referenced by TokenImpl::compareTo().

The documentation for this interface was generated from the following file:

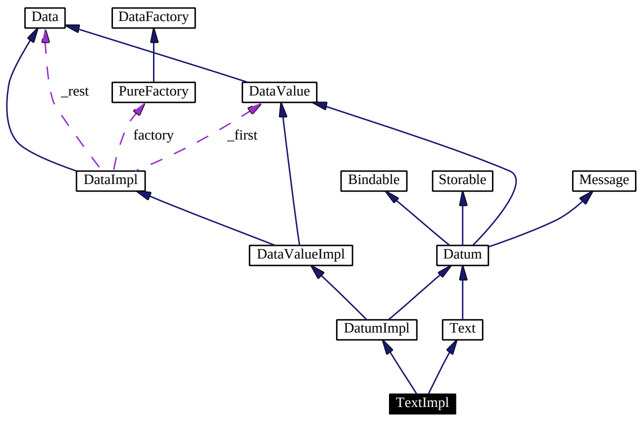
- Text.java

6.65 TextImpl Class Reference

Inheritance diagram for TextImpl:



Collaboration diagram for TextImpl:



Public Methods

- String stringValue ()
- Empty equals (Data data) throws Exceptional

Private Methods

- TextImpl (PureFactory factory, String string)

Private Attributes

- String string

6.65.1 Constructor & Destructor Documentation

6.65.1.1 TextImpl::TextImpl (PureFactory *factory*, String *string*) `[inline, private]`

Definition at line 9 of file TextImpl.java.

```
00009                                                  {
00010         super(factory);
00011         this.string = string;
00012     }
```

6.65.2 Member Function Documentation

6.65.2.1 Empty TextImpl::equals (Data *data*) `[inline]`

Reimplemented from DataValueImpl.

Definition at line 19 of file TextImpl.java.

```
00019                                                       {
00020         if (data instanceof Text) {
00021             if (string.compareTo(((Text)data).stringValue()) == 0)
00022                 return factory.makeEmpty();
00023         }
00024         throw new Exceptional(factory.makeEmpty());
00025     }
```

6.65.2.2 String TextImpl::stringValue () `[inline]`

Reimplemented from Text.

Definition at line 14 of file TextImpl.java.

```
00014                                 {
00015         return string;
00016     }
```

6.65.3 Member Data Documentation

6.65.3.1 String TextImpl::string `[private]`

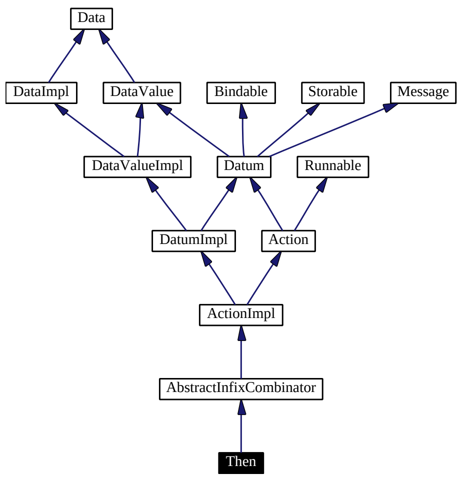
Definition at line 8 of file TextImpl.java.

The documentation for this class was generated from the following file:

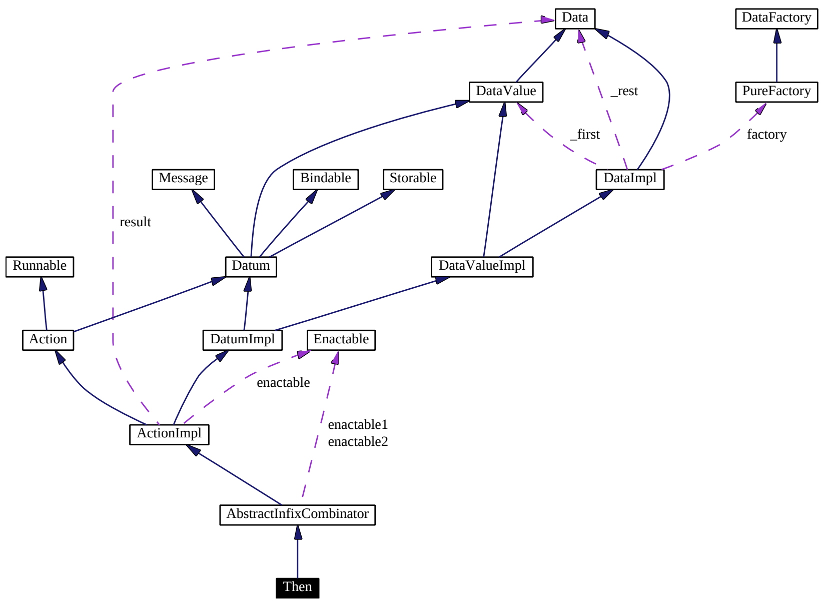
- TextImpl.java

6.66 Then Class Reference

Inheritance diagram for Then:



Collaboration diagram for Then:



Public Methods

- Then (PureFactory factory, Enactable enactable1, Enactable enactable2)
- Data enact (Data data, Bindings bindings) throws Exceptional, Failed
- Then (PureFactory factory, Enactable enactable1, Enactable enactable2)

6.66.1 Constructor & Destructor Documentation

6.66.1.1 Then::Then (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 64 of file AbstractAction.java.

```
                                                                                  {
           super(factory, enactable1, enactable2);
    }
```

6.66.1.2 Then::Then (PureFactory *factory*, Enactable *enactable1*, Enactable *enactable2*) `[inline]`

Definition at line 84 of file ActionImpl.java.

```
                                                                                  {
           super(factory, enactable1, enactable2);
           enactable = new _Then();
    }
```

6.66.2 Member Function Documentation

6.66.2.1 Data Then::enact (Data *data*, Bindings *bindings*) `[inline]`

Definition at line 67 of file AbstractAction.java.

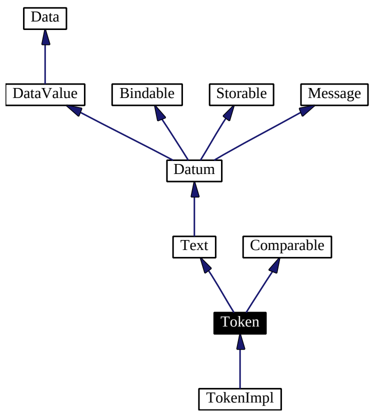
```
                                                                                {
           return enactable2.enact(enactable1.enact(data, bindings), bindings);
    }
```

The documentation for this class was generated from the following files:

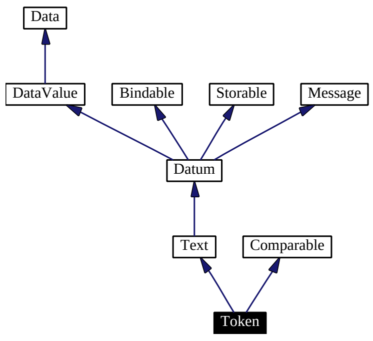
- AbstractAction.java
- ActionImpl.java

6.67 Token Interface Reference

Inheritance diagram for Token:



Collaboration diagram for Token:

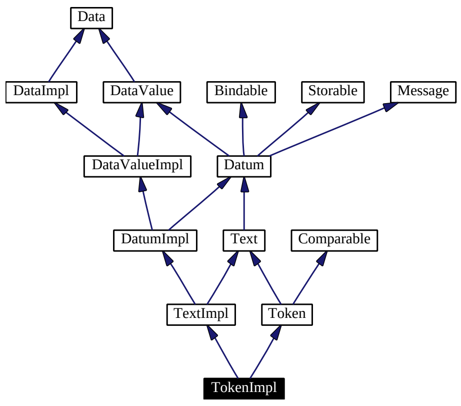


The documentation for this interface was generated from the following file:

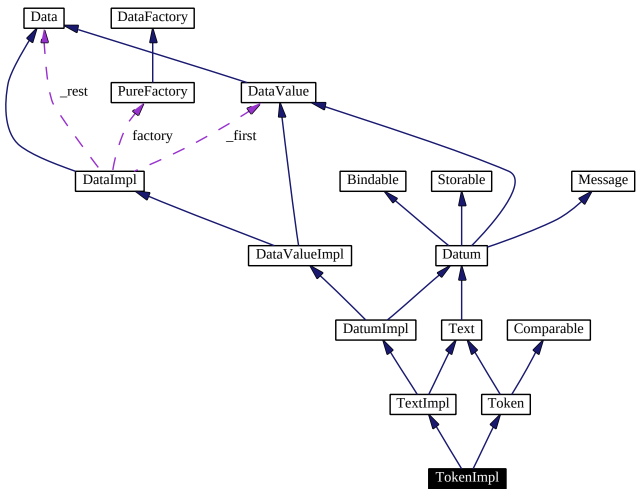
- Token.java

6.68 TokenImpl Class Reference

Inheritance diagram for TokenImpl:



Collaboration diagram for TokenImpl:



Public Methods

- int compareTo (Object o)

Private Methods

- TokenImpl (PureFactory factory, String string)

6.68.1 Constructor & Destructor Documentation

6.68.1.1 TokenImpl::TokenImpl (PureFactory *factory*, String *string*) `[inline, private]`

Definition at line 8 of file TokenImpl.java.

```
                                                  {
              super(factory, string);
          }
```

6.68.2 Member Function Documentation

6.68.2.1 int TokenImpl::compareTo (Object *o*) [inline]

Definition at line 12 of file TokenImpl.java.

```
                                   {
              return stringValue().compareTo(((Token)o).stringValue());
          }
```

The documentation for this class was generated from the following file:

- TokenImpl.java

Chapter 7

File Documentation

7.1 AbstractAction.java File Reference

Compounds

- class ActionImpl
- class And
- class AndExceptionally
- class AndThen
- class Exceptionally
- class Hence
- class Indivisibly
- class Otherwise
- class Then

7.2 AbstractEnactable.java File Reference

Compounds

- class AbstractEnactable

7.3 AbstractInfixCombinator.java File Reference

Compounds

- class AbstractInfixCombinator

7.4 AbstractPrefixCombinator.java File Reference

Compounds

- class AbstractPrefixCombinator

7.5 Action.java File Reference

Compounds

- interface Action

7.6 ActionImpl.java File Reference

Compounds

- class And::_And
- class AndExceptionally::_AndExceptionally
- class AndThen::_AndThen
- class Exceptionally::_Exceptionally
- class Hence::_Hence
- class Indivisibly::_Indivisibly
- class Otherwise::_Otherwise
- class Then::_Then
- class ActionImpl
- class And
- class AndExceptionally
- class AndThen
- class Exceptionally
- class Hence
- class Indivisibly
- class Otherwise
- class Then

7.7 Agent.java File Reference

Compounds

- interface Agent

7.8 AgentImpl.java File Reference

Compounds

- class AgentImpl

7.9 Bindable.java File Reference

Compounds

- interface Bindable

7.10 Bindings.java File Reference

Compounds

- interface Bindings

7.11 BindingsImpl.java File Reference

Compounds

- class BindingsImpl

7.12 Bool.java File Reference

Compounds

- interface Bool

7.13 BoolImpl.java File Reference

Compounds

- class BoolImpl

7.14 Cell.java File Reference

Compounds

- interface Cell

7.15 CellImpl.java File Reference

Compounds

- class CellImpl

7.16 Data.java File Reference

Compounds

- interface Data

7.17 DataConst.java File Reference

Compounds

- interface DataConst

7.18 DataConstImpl.java File Reference

Compounds

- class DataConstImpl

7.19 DataFactory.java File Reference

Compounds

- interface DataFactory

7.20 DataImpl.java File Reference

Compounds

- class Provide::_Provide
- class DataImpl
- class Provide

7.21 DataValue.java File Reference

Compounds

- interface DataValue

7.22 DataValueImpl.java File Reference

Compounds

- class DataValueImpl

7.23 Datum.java File Reference

Compounds

- interface Datum

7.24 DatumImpl.java File Reference

Compounds

- class DatumImpl

7.25 Empty.java File Reference

Compounds

- interface Empty

7.26 EmptyImpl.java File Reference

Compounds

- class EmptyImpl

7.27 Enactable.java File Reference

Compounds

- interface Enactable

7.28 Exceptional.java File Reference

Compounds

- class Exceptional

7.29 Failed.java File Reference

Compounds

- class Failed

7.30 Int.java File Reference

Compounds

- interface Int

7.31 IntImpl.java File Reference

Compounds

- class IntImpl

7.32 List.java File Reference

Compounds

- interface List

7.33 ListImpl.java File Reference

Compounds

- class ListImpl

7.34 Message.java File Reference

Compounds

- interface Message

7.35 MessageTag.java File Reference

Compounds

- interface MessageTag

7.36 MessageTagImpl.java File Reference

Compounds

- class MessageTagImpl

7.37 PureFactory.java File Reference

Compounds

- class PureFactory

7.38 Schedule.java File Reference

Compounds

- class Schedule

7.39 Scheduling.java File Reference

Compounds

- interface Scheduling

7.40 Storable.java File Reference

Compounds

- interface Storable

7.41 Store.java File Reference

Compounds

- class Store

7.42 Storing.java File Reference

Compounds

- interface Storing

7.43 TaggedBuffers.java File Reference

Compounds

- class TaggedBuffers

7.44 Text.java File Reference

Compounds

- interface Text

7.45 TextImpl.java File Reference

Compounds

- class TextImpl

7.46 Token.java File Reference

Compounds

- interface Token

7.47 TokenImpl.java File Reference

Compounds

- class TokenImpl

7.48 Tuple.java File Reference