Introduction to ImgLib2

Stephan Preibisch, Tobias Pietzsch, Stephan Saalfeld

Albert Einstein College of Medicine, New York
MPI-CBG, Dresden, Germany
HHMI Janelia Farm Research Campus, Virginia

Albert Einstein College of Medicine OF YESHIVA UNIVERSITY

janelia farm research campus

CBG Max Planck Institute of Molecular Cell Biology and Genetics

Special thanks to …

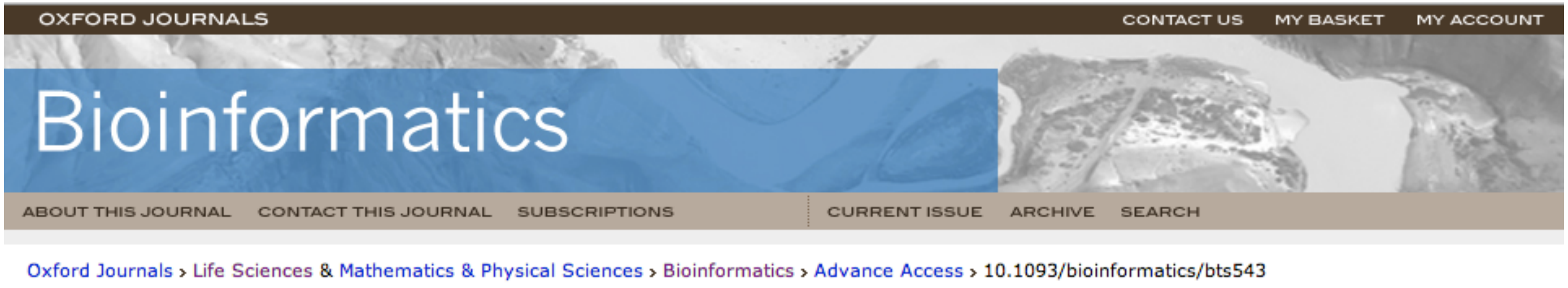
- <u>Tobias Pietzsch (MPI-CBG)</u>
- <u>Stephan Saalfeld (MPI-CBG)</u>

- Pavel Tomancak (MPI-CBG)
- Gene Myers (MPI-CBG)
- Rob Singer (Einstein College & Janelia Farm)

- ImageJ2 crew
  - Johannes Schindelin
  - Curtis Rueden
  - Barry DeZonia
  - Kevin Eliceiri
- Albert Cardona
- KNIME guys
  - Christian Dietz
  - Martin Horn

HFSP

ImgLib2



OXFORD JOURNALS

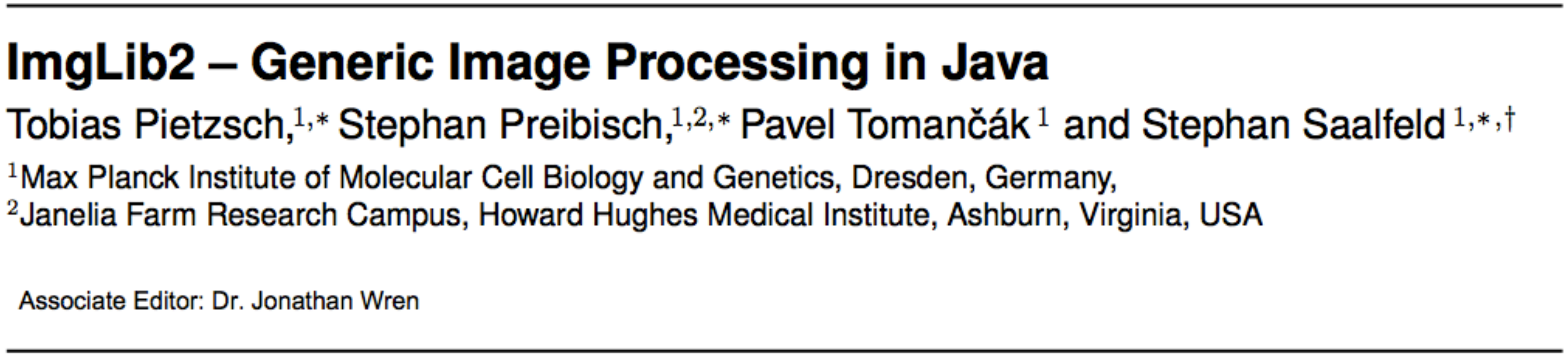
CONTACT US MY BASKET MY ACCOUNT

Bioinformatics

ABOUT THIS JOURNAL CONTACT THIS JOURNAL SUBSCRIPTIONS CURRENT ISSUE ARCHIVE SEARCH

Oxford Journals > Life Sciences & Mathematics & Physical Sciences > Bioinformatics > Advance Access > 10.1093/bioinformatics/bts543

ImgLib2 paper is finally out …



ImgLib2 – Generic Image Processing in Java

Tobias Pietzsch,[1,*] Stephan Preibisch,[1,2,*] Pavel Tomančák[1] and Stephan Saalfeld[1,*,†]

[1]Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany,
[2]Janelia Farm Research Campus, Howard Hughes Medical Institute, Ashburn, Virginia, USA

Associate Editor: Dr. Jonathan Wren

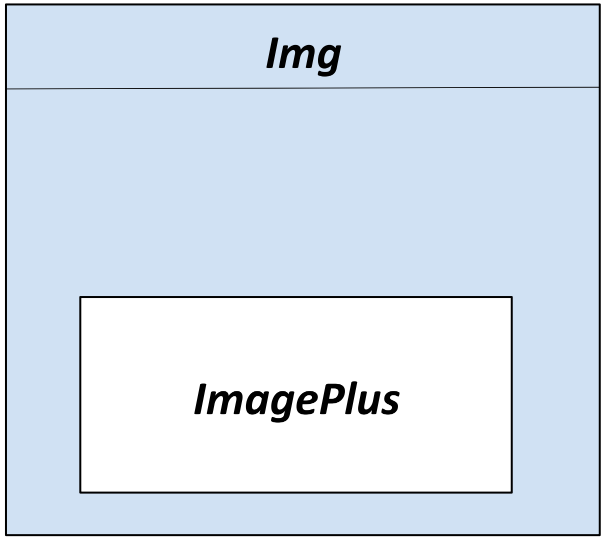
ImgLib2

Why using ImgLib2?

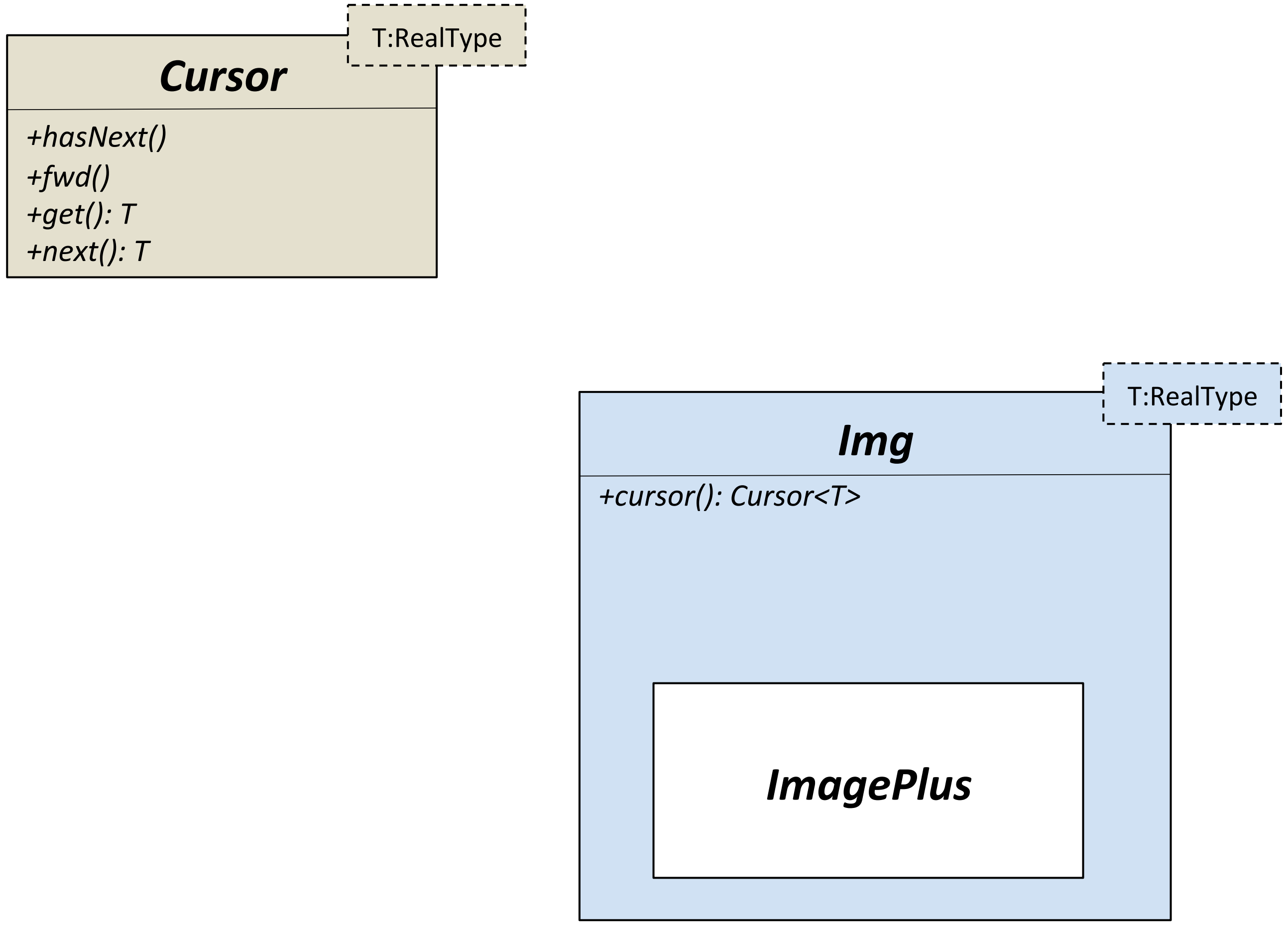
- Generic programming – Write it once!
- Directly applicable to very large datasets, different dimensionality or pixel type
- ImgLib2 does not force you to implement type independent or n-dimensional – it is still applicable to very large datasets
- More algorithm-like programming
- Fewer simple programming mistakes
- Easier exchange of code
- Smaller source code

ImgLib2

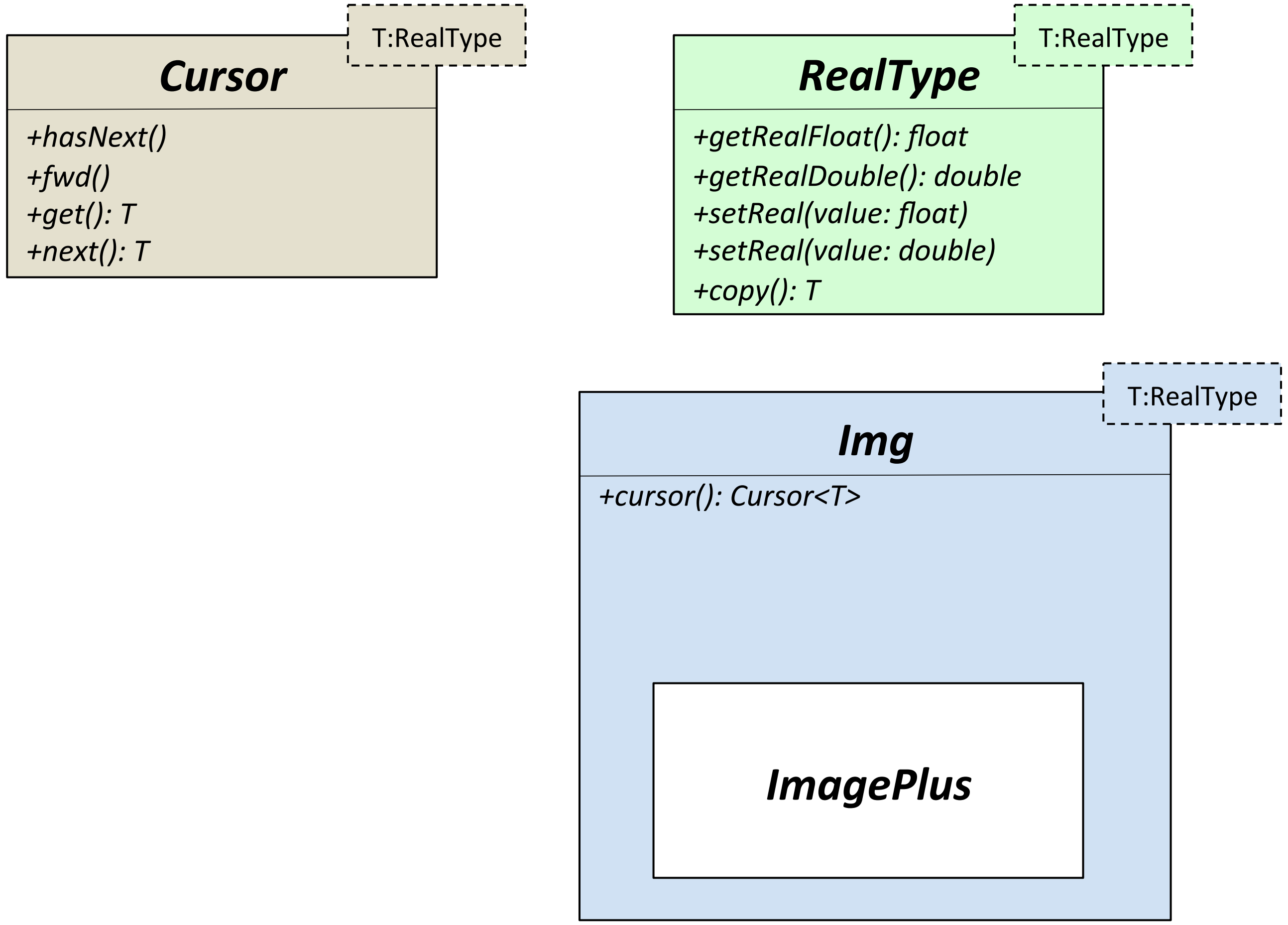
ImagePlus



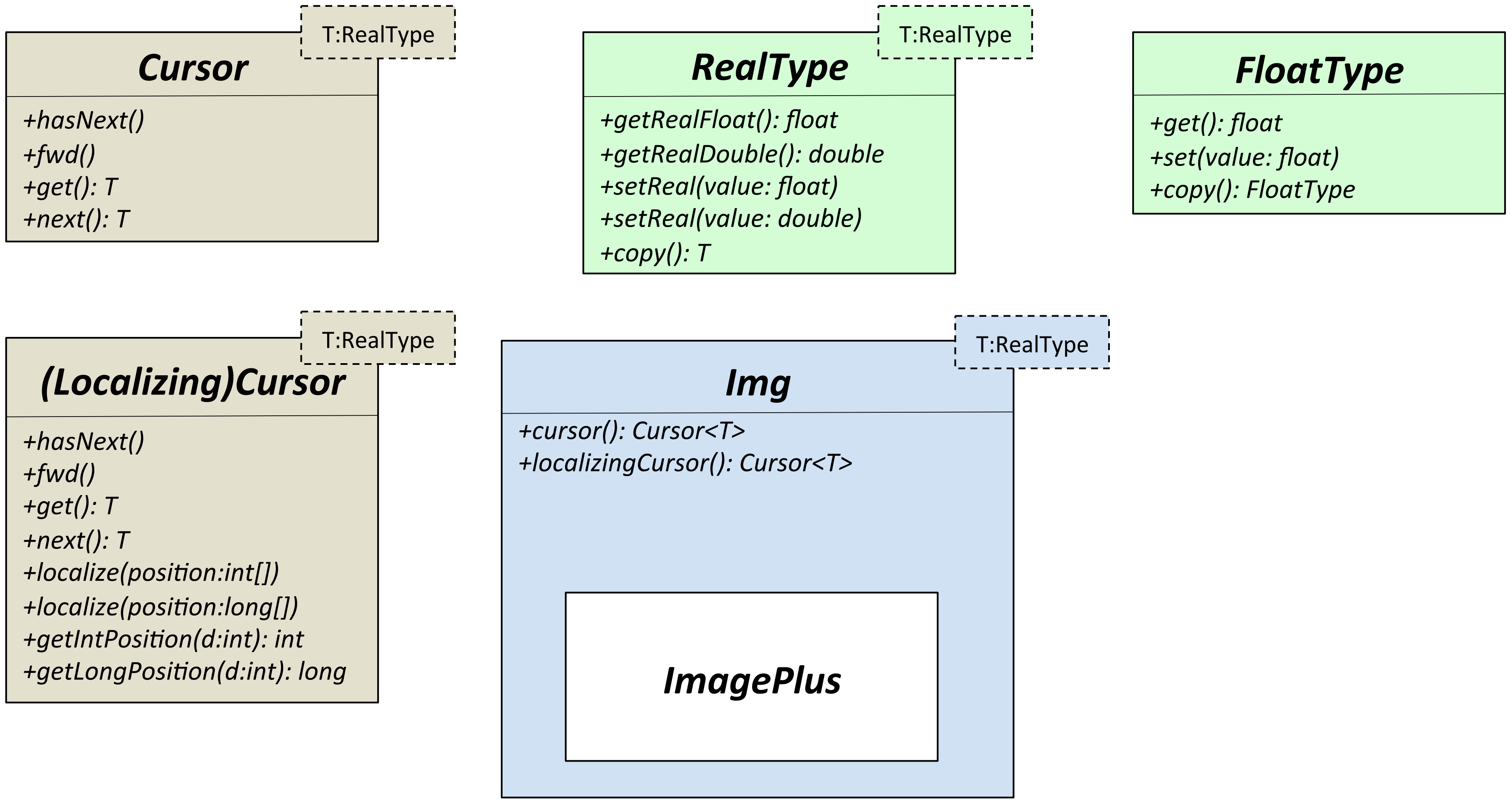
ImgLib2


T:RealType
Cursor
+hasNext()
+fwd()
+get(): T
+next(): T
T:RealType
Img
+cursor(): Cursor<T>
ImagePlus

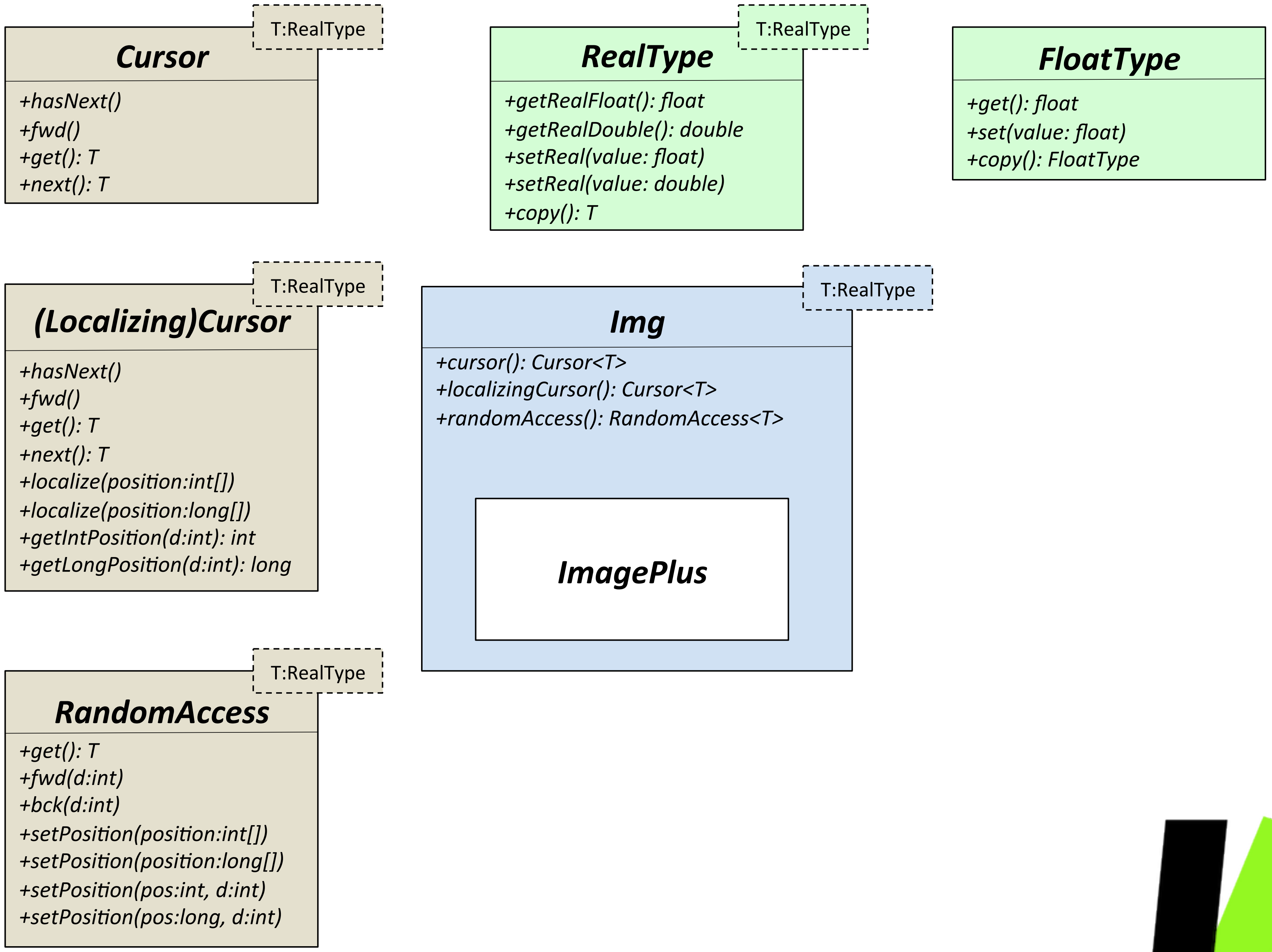
ImgLib2


Cursor
T:RealType
+hasNext()
+fwd()
+get(): T
+next(): T
RealType
T:RealType
+getRealFloat(): float
+getRealDouble(): double
+setReal(value: float)
+setReal(value: double)
+copy(): T
FloatType
+get(): float
+set(value: float)
+copy(): FloatType
Img
T:RealType
+cursor(): Cursor<T>
ImagePlus

ImgLib2


T:RealType
Cursor
+hasNext()
+fwd()
+get(): T
+next(): T
T:RealType
RealType
+getRealFloat(): float
+getRealDouble(): double
+setReal(value: float)
+setReal(value: double)
+copy(): T
FloatType
+get(): float
+set(value: float)
+copy(): FloatType
T:RealType
(Localizing)Cursor
+hasNext()
+fwd()
+get(): T
+next(): T
+localize(position:int[])
+localize(position:long[])
+getIntPosition(d:int): int
+getLongPosition(d:int): long
T:RealType
Img
+cursor(): Cursor<T>
+localizingCursor(): Cursor<T>
ImagePlus

ImgLib2



T:RealType
Cursor
- _+hasNext()_
- _+fwd()_
- _+get(): T_
- _+next(): T_

T:RealType
RealType
- _+getRealFloat(): float_
- _+getRealDouble(): double_
- _+setReal(value: float)_
- _+setReal(value: double)_
- _+copy(): T_

FloatType
- _+get(): float_
- _+set(value: float)_
- _+copy(): FloatType_

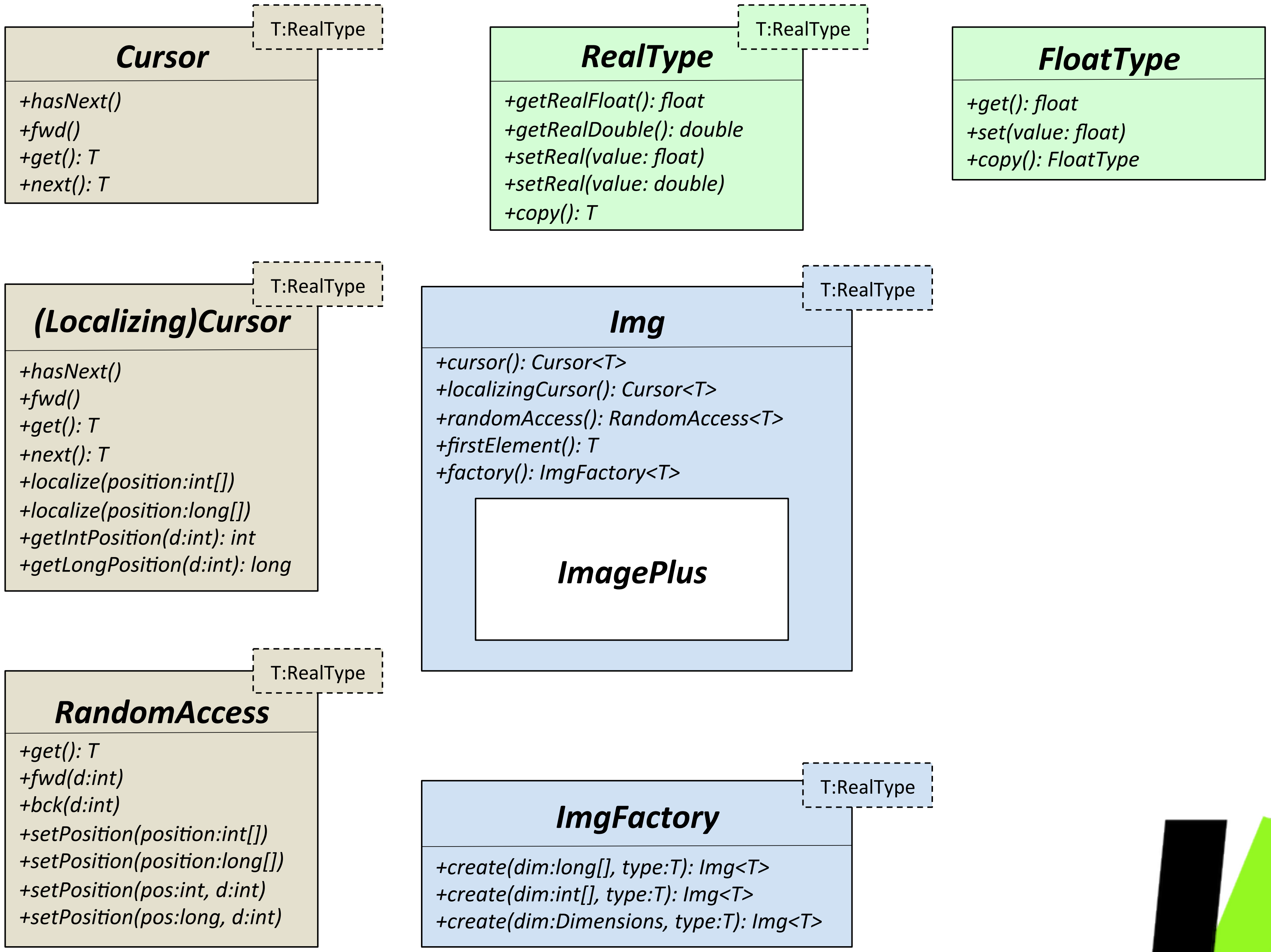
T:RealType
(Localizing)Cursor
- _+hasNext()_
- _+fwd()_
- _+get(): T_
- _+next(): T_
- _+localize(position:int[])_
- _+localize(position:long[])_
- _+getIntPosition(d:int): int_
- _+getLongPosition(d:int): long_

T:RealType
Img
- _+cursor(): Cursor<T>_
- _+localizingCursor(): Cursor<T>_
- _+randomAccess(): RandomAccess<T>_

ImagePlus

T:RealType
RandomAccess
- _+get(): T_
- _+fwd(d:int)_
- _+bck(d:int)_
- _+setPosition(position:int[])_
- _+setPosition(position:long[])_
- _+setPosition(pos:int, d:int)_
- _+setPosition(pos:long, d:int)_

ImgLib2



Cursor T:RealType

```
+hasNext()
+fwd()
+get(): T
+next(): T
```

RealType T:RealType

```
+getRealFloat(): float
+getRealDouble(): double
+setReal(value: float)
+setReal(value: double)
+copy(): T
```

FloatType

```
+get(): float
+set(value: float)
+copy(): FloatType
```

(Localizing)Cursor T:RealType

```
+hasNext()
+fwd()
+get(): T
+next(): T
+localize(position:int[])
+localize(position:long[])
+getIntPosition(d:int): int
+getLongPosition(d:int): long
```

Img T:RealType

```
+cursor(): Cursor<T>
+localizingCursor(): Cursor<T>
+randomAccess(): RandomAccess<T>
+firstElement(): T
+factory(): ImgFactory<T>
```

ImagePlus

RandomAccess T:RealType

```
+get(): T
+fwd(d:int)
+bck(d:int)
+setPosition(position:int[])
+setPosition(position:long[])
+setPosition(pos:int, d:int)
+setPosition(pos:long, d:int)
```

ImgFactory T:RealType

```
+create(dim:long[], type:T): Img<T>
+create(dim:int[], type:T): Img<T>
+create(dim:Dimensions, type:T): Img<T>
```

ImgLib2

T:RealType

Cursor

+hasNext()
+fwd()
+get(): T
+next(): T

T:RealType

(Localizing)Cursor

+hasNext()
+fwd()
+get(): T
+next(): T
+localize(position:int[])
+localize(position:long[])
+getIntPosition(d:int): int
+getLongPosition(d:int): long

T:RealType

RandomAccess

+get(): T
+fwd(d:int)
+bck(d:int)
+setPosition(position:int[])
+setPosition(position:long[])
+setPosition(pos:int, d:int)
+setPosition(pos:long, d:int)

T:RealType

RealType

+getRealFloat(): float
+getRealDouble(): double
+setReal(value: float)
+setReal(value: double)
+copy(): T

T:RealType

Img

+cursor(): Cursor<T>
+localizingCursor(): Cursor<T>
+randomAccess(): RandomAccess<T>
+firstElement(): T
+factory(): ImgFactory<T>

ImagePlus

T:RealType

ImgFactory

+create(dim:long[], type:T): Img<T>
+create(dim:int[], type:T): Img<T>
+create(dim:Dimensions, type:T): Img<T>

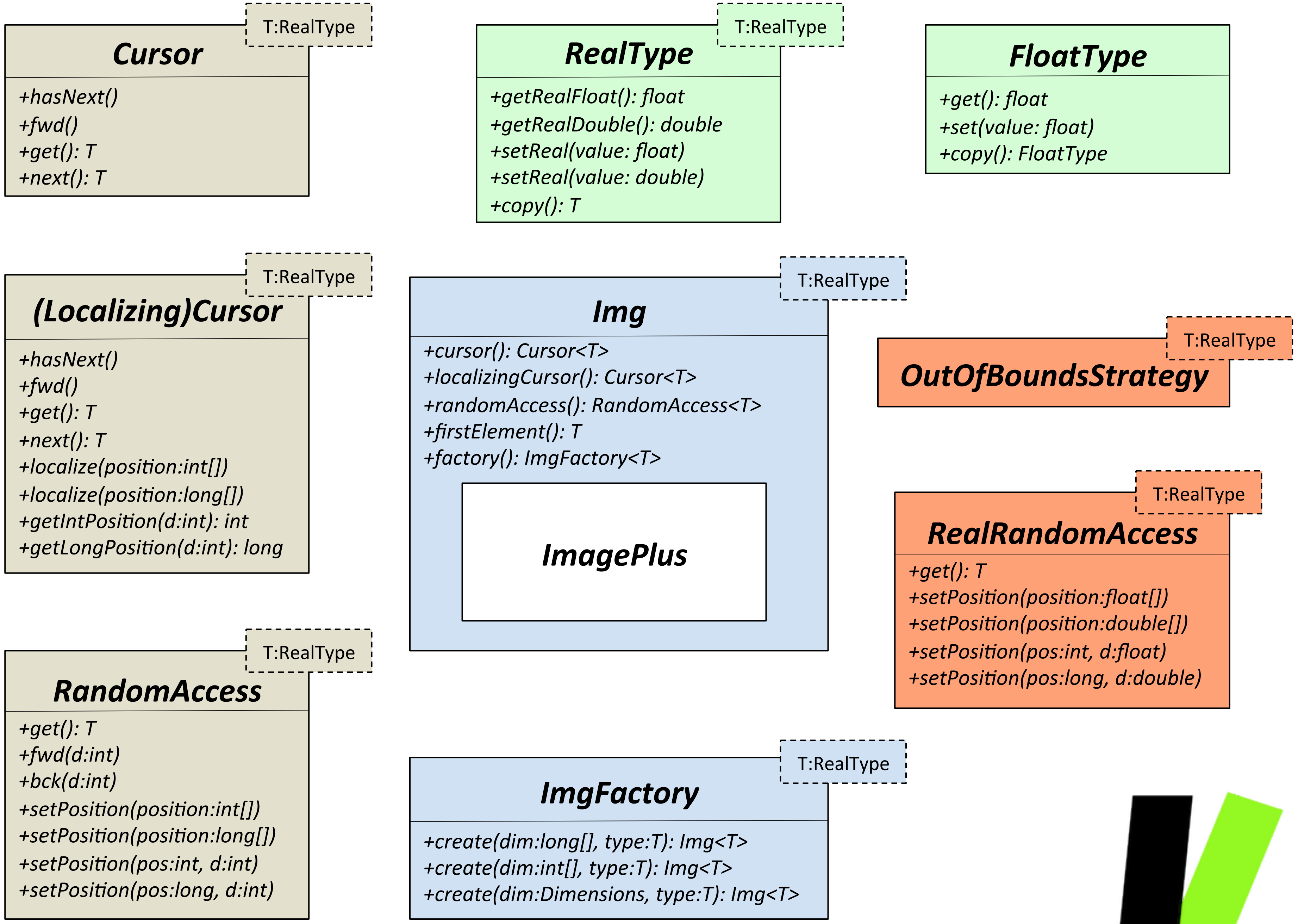
FloatType

+get(): float
+set(value: float)
+copy(): FloatType

T:RealType

OutOfBoundsStrategy

ImgLib2



Cursor (T:RealType)
- +hasNext()
- +fwd()
- +get(): T
- +next(): T

RealType (T:RealType)
- +getRealFloat(): float
- +getRealDouble(): double
- +setReal(value: float)
- +setReal(value: double)
- +copy(): T

FloatType
- +get(): float
- +set(value: float)
- +copy(): FloatType

(Localizing)Cursor (T:RealType)
- +hasNext()
- +fwd()
- +get(): T
- +next(): T
- +localize(position:int[])
- +localize(position:long[])
- +getIntPosition(d:int): int
- +getLongPosition(d:int): long

Img (T:RealType)
- +cursor(): Cursor<T>
- +localizingCursor(): Cursor<T>
- +randomAccess(): RandomAccess<T>
- +firstElement(): T
- +factory(): ImgFactory<T>

ImagePlus

OutOfBoundsStrategy (T:RealType)

RealRandomAccess (T:RealType)
- +get(): T
- +setPosition(position:float[])
- +setPosition(position:double[])
- +setPosition(pos:int, d:float)
- +setPosition(pos:long, d:double)

RandomAccess (T:RealType)
- +get(): T
- +fwd(d:int)
- +bck(d:int)
- +setPosition(position:int[])
- +setPosition(position:long[])
- +setPosition(pos:int, d:int)
- +setPosition(pos:long, d:int)

ImgFactory (T:RealType)
- +create(dim:long[], type:T): Img<T>
- +create(dim:int[], type:T): Img<T>
- +create(dim:Dimensions, type:T): Img<T>

ImgLib2

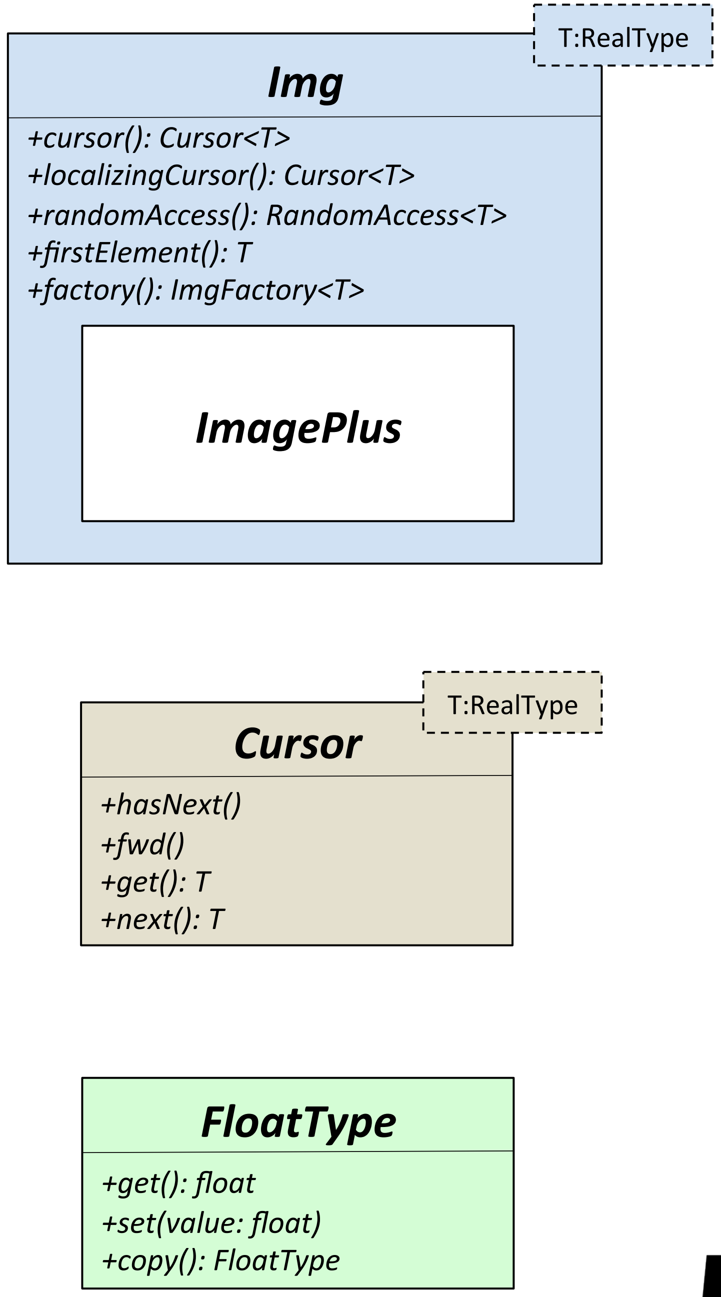
Hands on Programming

① Threshold on an image
② Center of mass of an image
③ Gradient of an image
④ Rigid transformation of an image

ImgLib2

Thresholding 1

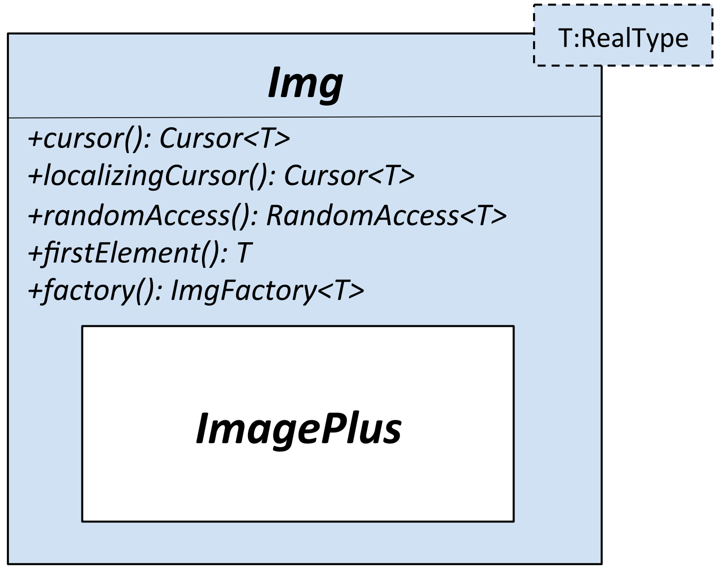
- Wrap a float image
- Compute threshold for each pixel
- Overwrite the original data
- ImgLib2_Threshold1.java

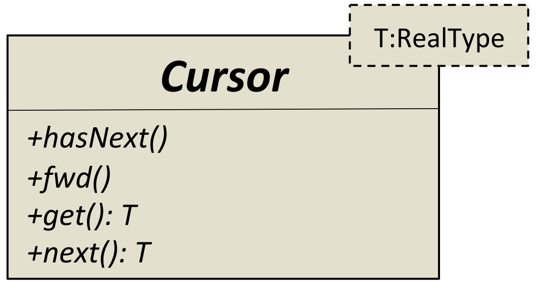


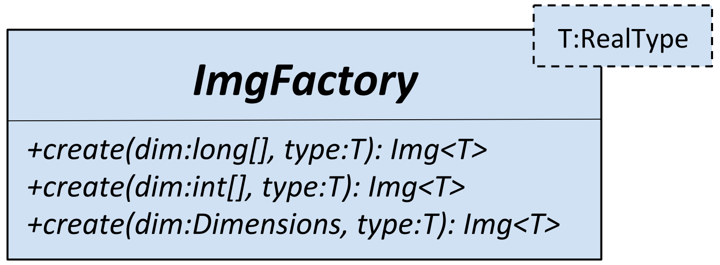
ImgLib2

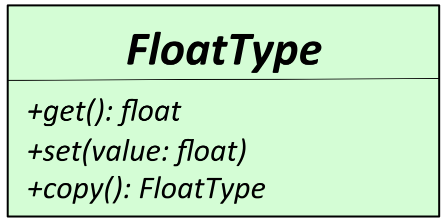
Thresholding 2

- Wrap a float image
- Create a new Img
- Compute threshold for each pixel
- Write threshold into the new Img
- ImgLib2_Threshold2.java


T:RealType
Img
+cursor(): Cursor<T>
+localizingCursor(): Cursor<T>
+randomAccess(): RandomAccess<T>
+firstElement(): T
+factory(): ImgFactory<T>
ImagePlus



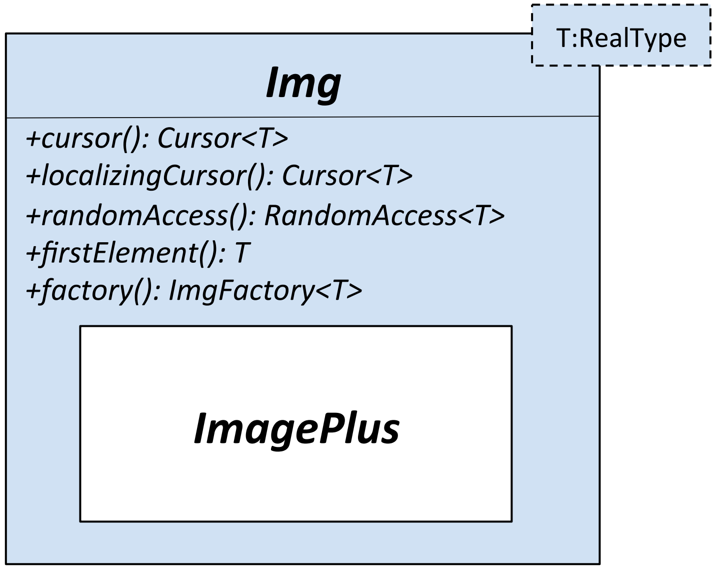

T:RealType
ImgFactory
+create(dim:long[], type:T): Img<T>
+create(dim:int[], type:T): Img<T>
+create(dim:Dimensions, type:T): Img<T>

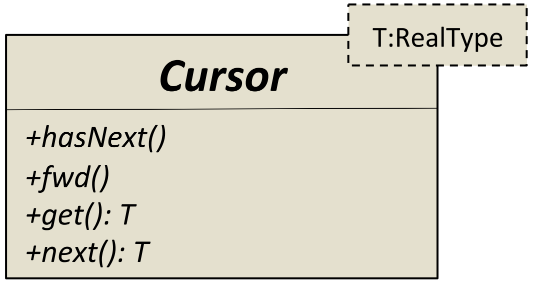


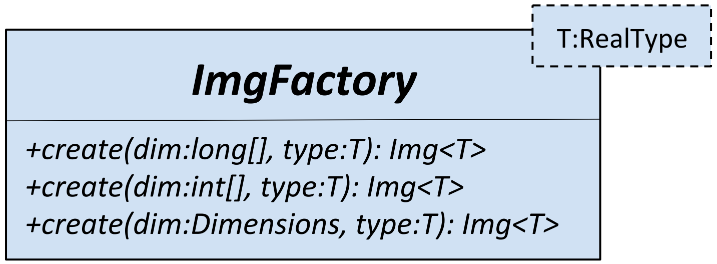
ImgLib2

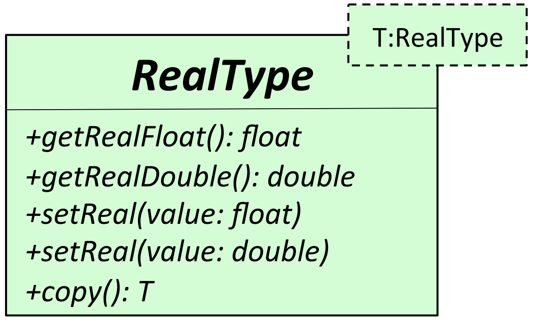
Thresholding 3

- Wrap <u>any real valued</u> image
- Create a new Img
- Compute threshold for each pixel
- Write threshold into the new Img
- ImgLib2_Threshold3.java


T:RealType
Img
+cursor(): Cursor<T>
+localizingCursor(): Cursor<T>
+randomAccess(): RandomAccess<T>
+firstElement(): T
+factory(): ImgFactory<T>
ImagePlus

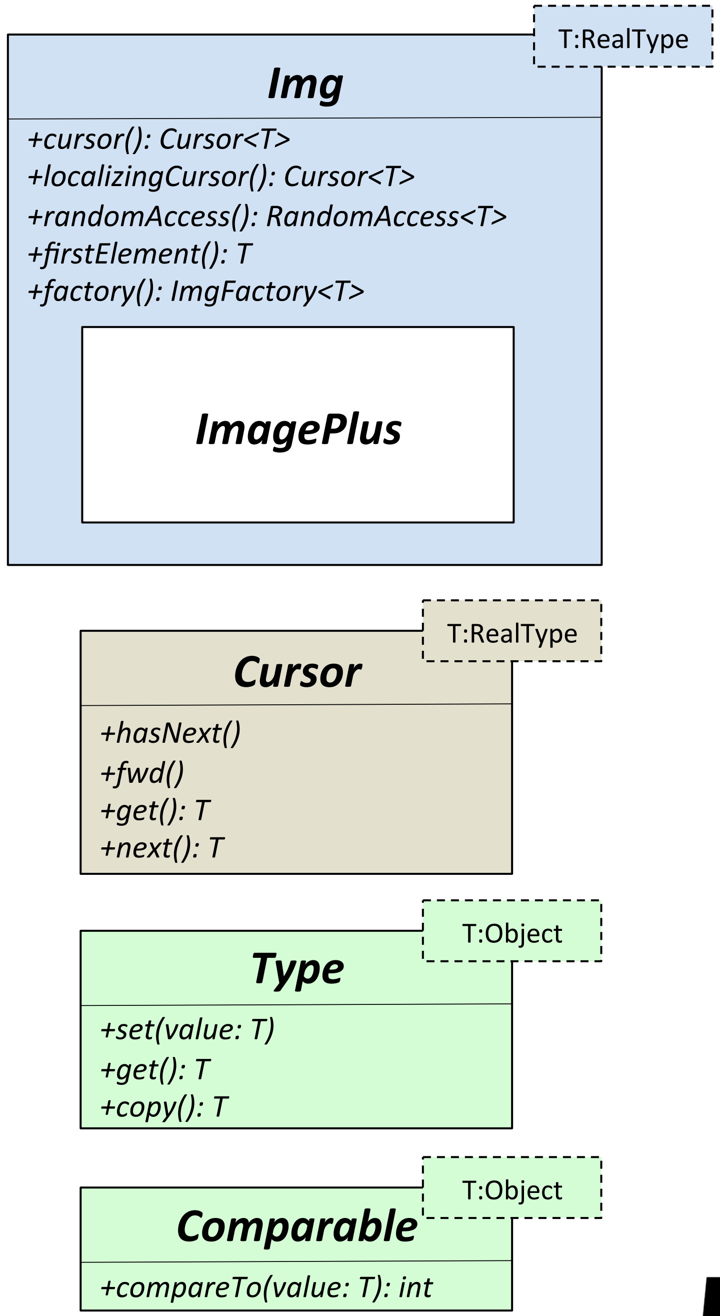


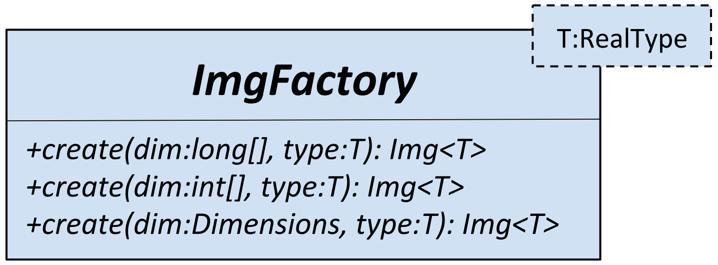

T:RealType
ImgFactory
+create(dim:long[], type:T): Img<T>
+create(dim:int[], type:T): Img<T>
+create(dim:Dimensions, type:T): Img<T>



Thresholding 4

- Wrap any real valued image
- Create a new Img
- Compute threshold for each pixel for all <u>Comparable</u>
- Write threshold into the new <u>Img of BitType (1 bit per pixel)</u>
- ImgLib2_Threshold4.java

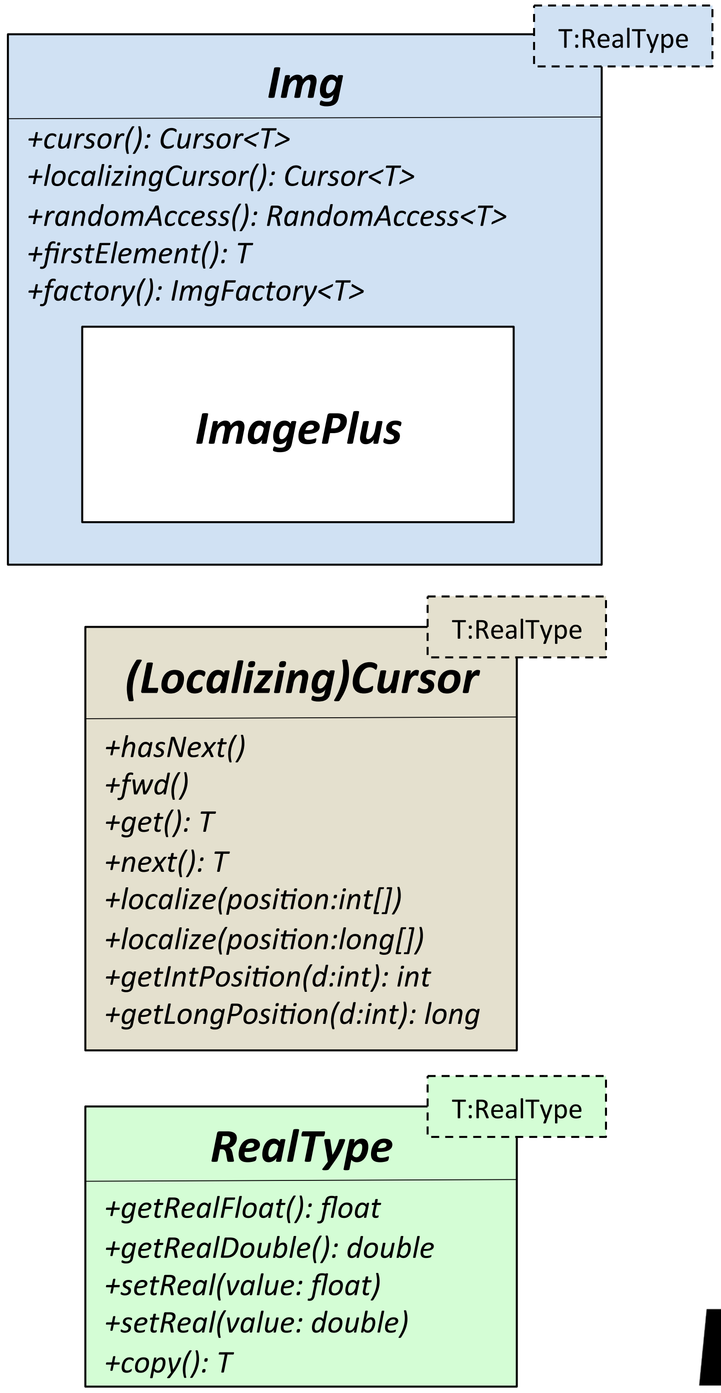



T:RealType
ImgFactory
+create(dim:long[], type:T): Img<T>
+create(dim:int[], type:T): Img<T>
+create(dim:Dimensions, type:T): Img<T>

ImgLib2

Center of Mass 1

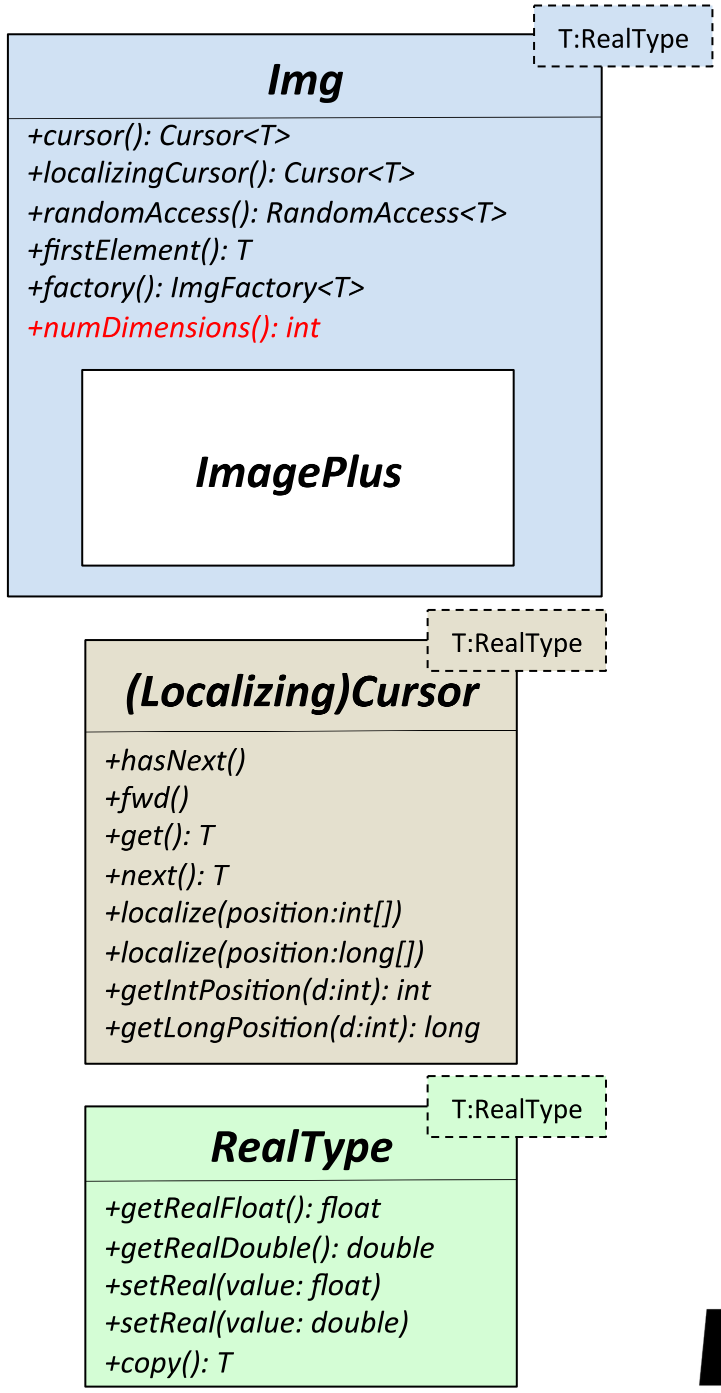
- Wrap any real valued image
- Use a localizing Cursor to Compute the center of mass in two dimensions (x,y)
- Write the result to the log window
- ImgLib2_CenterOfMass1.java



ImgLib2

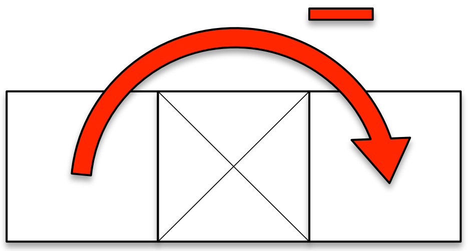
Center of Mass 2

- Wrap any real valued image
- Use a localizing Cursor to Compute the center of mass in <u>n dimensions</u>
- Write the result to the log window
- ImgLib2_CenterOfMass2.java



Gradient 1

- Wrap any real valued image
- Approximate the magnitude of the gradient for each pixel using a localizing Cursor on the output and a RandomAccess on the input



$$\nabla f_x = \frac{I(x+1,y,\ldots) - I(x-1,y,\ldots)}{2}$$

$$\nabla f_y = \frac{I(x,y+1,\ldots) - I(x,y-1,\ldots)}{2}$$

$$|\nabla f| = \sqrt{\nabla f_x^2 + \nabla f_y^2 + \ldots}$$

- ImgLib2_Gradient1.java

T:RealType

Img

+cursor(): Cursor<T>
+localizingCursor(): Cursor<T>
+randomAccess(): RandomAccess<T>
+firstElement(): T
+factory(): ImgFactory<T>
+numDimensions(): int
+dimension(d:int): long
+min(d:int): long
+max(d:int): long

ImagePlus

T:RealType

RealType

+getRealFloat(): float
+getRealDouble(): double
+setReal(value: float)
+setReal(value: double)
+copy(): T

T:RealType

(Localizing)Cursor

+hasNext()
+fwd()
+get(): T
+next(): T
+localize(position:int[])
+localize(position:long[])
+getIntPosition(d:int): int
+getLongPosition(d:int): long

T:RealType

RandomAccess

+get(): T
+fwd(d:int)
+bck(d:int)
+setPosition(position:int[])
+setPosition(position:long[])
+setPosition(pos:int, d:int)
+setPosition(pos:long, d:int)

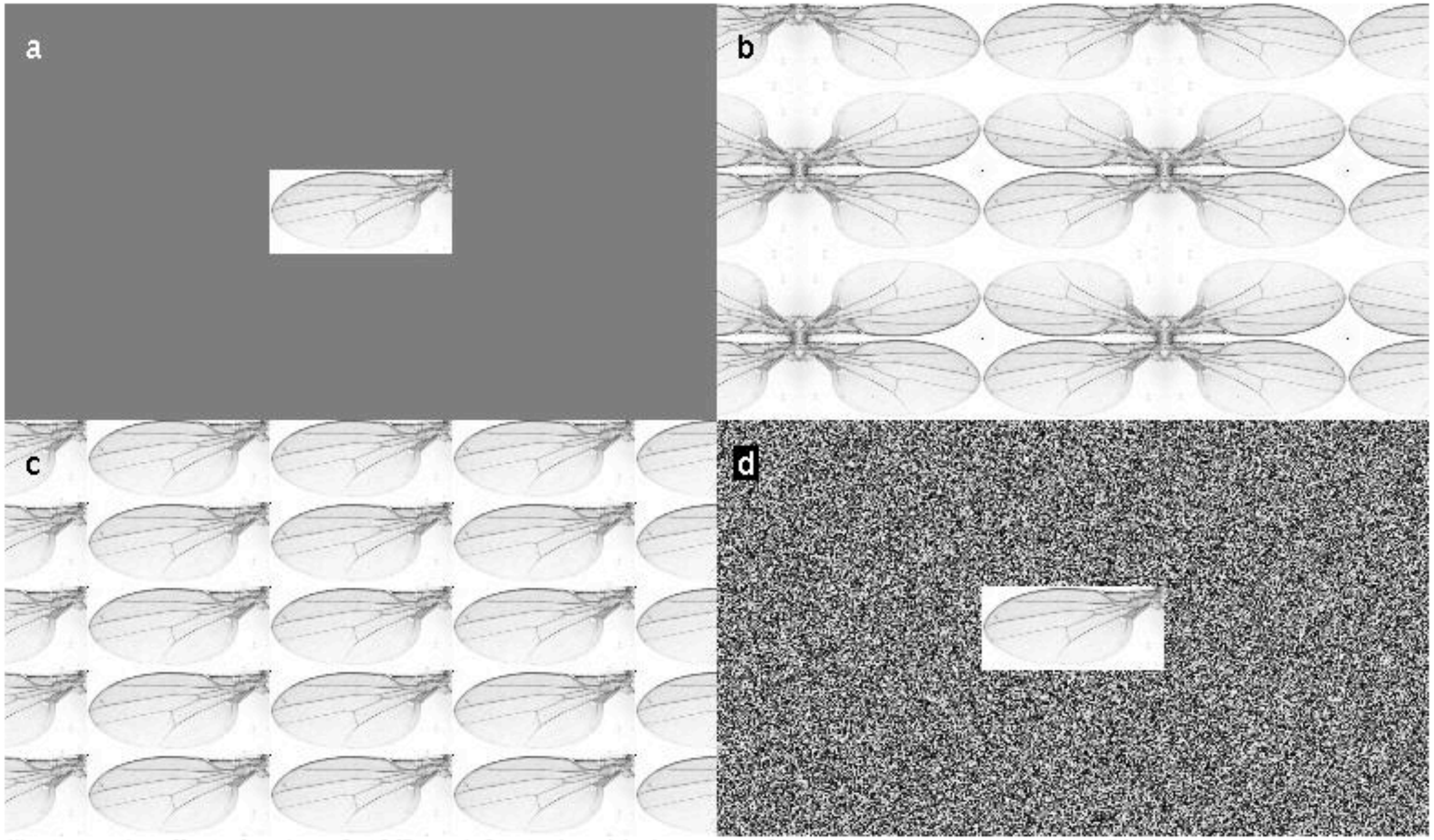
T:RealType

ImgFactory

+create(dim:long[], type:T): Img<T>
+create(dim:int[], type:T): Img<T>
+create(dim:Dimensions, type:T): Img<T>

ImgLib2

OutOfBoundsStrategies

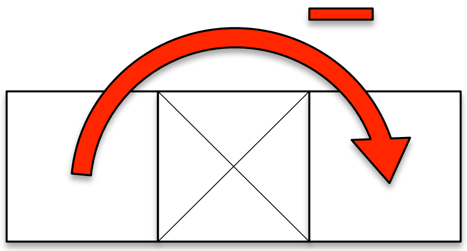


Illustrates the effect of various OutOfBoundsStrategies. (a) shows out of bounds with a constant value, (b) shows a mirroring strategy, (c) shows the periodic strategy, and (d) shows a strategy that uses random values.

ImgLib2

Gradient 2

- Use OutOfBoundsStrategy to compute gradient for all pixels
- Approximate the magnitude of the gradient for each pixel using a localizing Cursor on the output and a RandomAccess on the input



$$\nabla f_x = \frac{I(x+1,y,\ldots) - I(x-1,y,\ldots)}{2}$$

$$\nabla f_y = \frac{I(x,y+1,\ldots) - I(x,y-1,\ldots)}{2}$$

$$|\nabla f| = \sqrt{\nabla f_x^2 + \nabla f_y^2 + \ldots}$$

- ImgLib2_Gradient2.java

T:RealType

Img

+cursor(): Cursor<T>
+localizingCursor(): Cursor<T>
+randomAccess(): RandomAccess<T>
+firstElement(): T
+factory(): ImgFactory<T>
+numDimensions(): int
+dimension(d:int): long
+min(d:int): long
+max(d:int): long

ImagePlus

T:RealType

RealType

+getRealFloat(): float
+getRealDouble(): double
+setReal(value: float)
+setReal(value: double)
+copy(): T

T:RealType

(Localizing)Cursor

+hasNext()
+fwd()
+get(): T
+next(): T
+localize(position:int[])
+localize(position:long[])
+getIntPosition(d:int): int
+getLongPosition(d:int): long

T:RealType

RandomAccess

+get(): T
+fwd(d:int)
+bck(d:int)
+setPosition(position:int[])
+setPosition(position:long[])
+setPosition(pos:int, d:int)
+setPosition(pos:long, d:int)

T:RealType

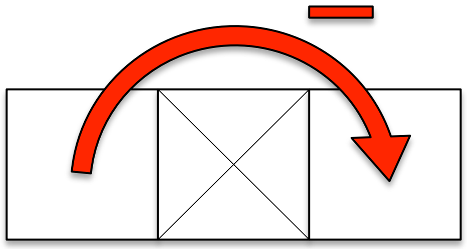
ImgFactory

+create(dim:long[], type:T): Img<T>
+create(dim:int[], type:T): Img<T>
+create(dim:Dimensions, type:T): Img<T>

ImgLib2

Gradient 3

- Always return an Img<FloatType> to prevent overflows
- Approximate the magnitude of the gradient for each pixel using a localizing Cursor on the output and a RandomAccess on the input



$$\nabla f_x = \frac{I(x+1,y,\dots) - I(x-1,y,\dots)}{2}$$

$$\nabla f_y = \frac{I(x,y+1,\dots) - I(x,y-1,\dots)}{2}$$

$$|\nabla f| = \sqrt{\nabla f_x^2 + \nabla f_y^2 + \dots}$$

- ImgLib2_Gradient3.java

Img (T:RealType)

+cursor(): Cursor<T>
+localizingCursor(): Cursor<T>
+randomAccess(): RandomAccess<T>
+firstElement(): T
+factory(): ImgFactory<T>
+numDimensions(): int
+dimension(d:int): long
+min(d:int): long
+max(d:int): long

ImagePlus

RealType (T:RealType)

+getRealFloat(): float
+getRealDouble(): double
+setReal(value: float)
+setReal(value: double)
+copy(): T

PlanarImgFactory (T:RealType)

+create(dim:long[], type:T): Img<T>
+create(dim:int[], type:T): Img<T>
+create(dim:Dimensions, type:T): Img<T>

(Localizing)Cursor (T:RealType)

+hasNext()
+fwd()
+get(): T
+next(): T
+localize(position:int[])
+localize(position:long[])
+getIntPosition(d:int): int
+getLongPosition(d:int): long

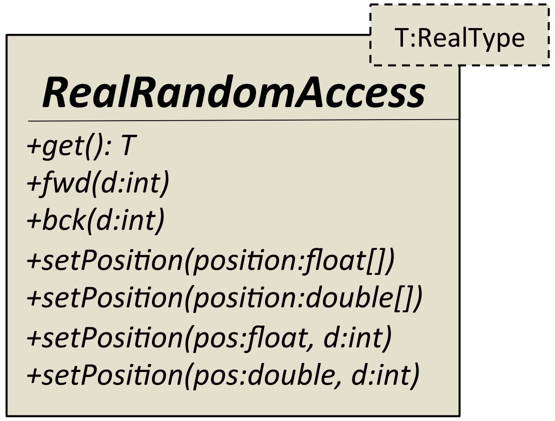
RandomAccess (T:RealType)

+get(): T
+fwd(d:int)
+bck(d:int)
+setPosition(position:int[])
+setPosition(position:long[])
+setPosition(pos:int, d:int)
+setPosition(pos:long, d:int)

ImgLib2

Interpolation

- Transform an Img (RandomAccessible) into a RealRandomAccessible that can return values at any real-valued location in space
- First extend by OutOfBounds



T:RealType

RealRandomAccess

+get(): T
+fwd(d:int)
+bck(d:int)
+setPosition(position:float[])
+setPosition(position:double[])
+setPosition(pos:float, d:int)
+setPosition(pos:double, d:int)

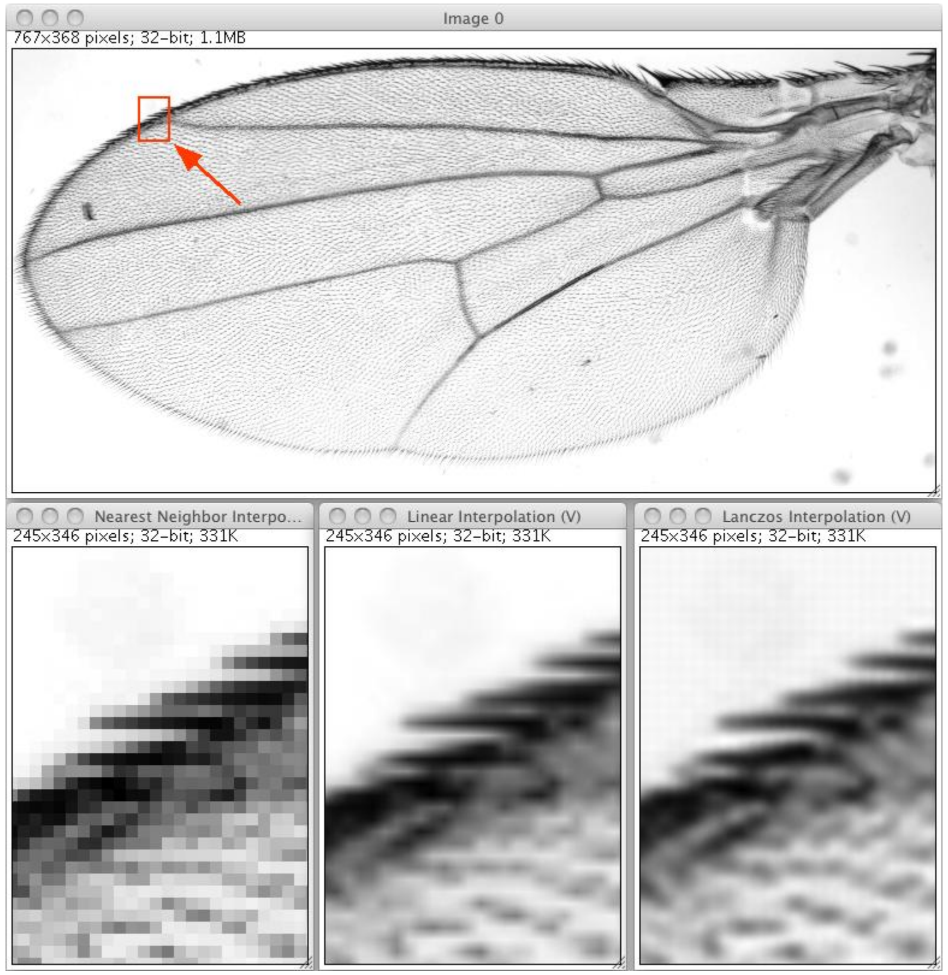
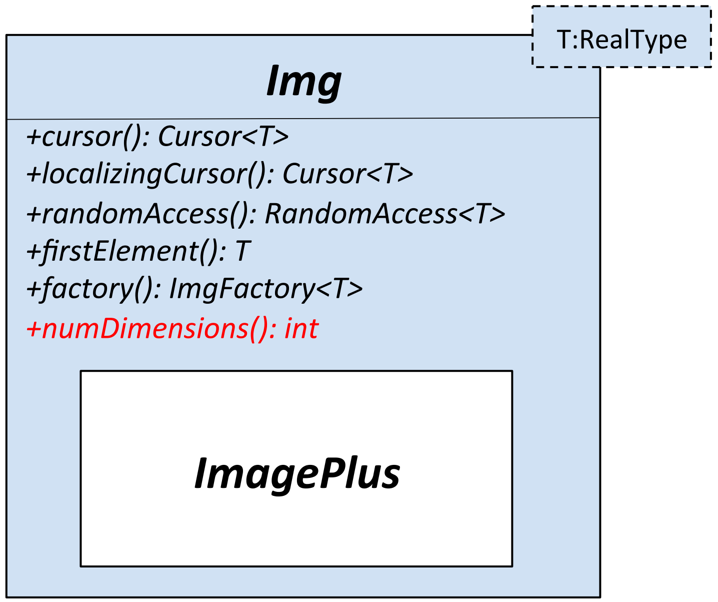


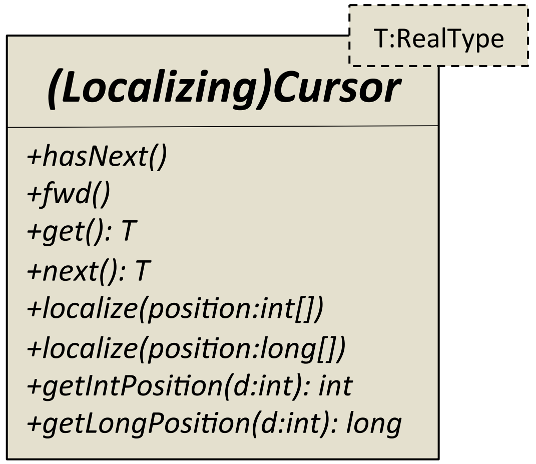
Image 0
767x368 pixels; 32-bit; 1.1MB
Nearest Neighbor Interpo...
245x346 pixels; 32-bit; 331K
Linear Interpolation (V)
245x346 pixels; 32-bit; 331K
Lanczos Interpolation (V)
245x346 pixels; 32-bit; 331K

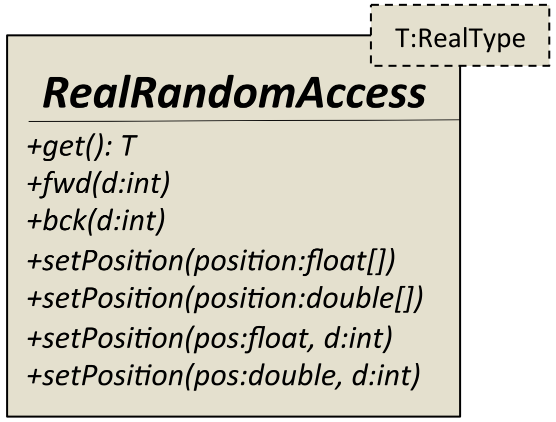
ImgLib2

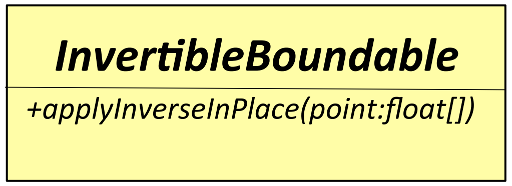
Image Transform

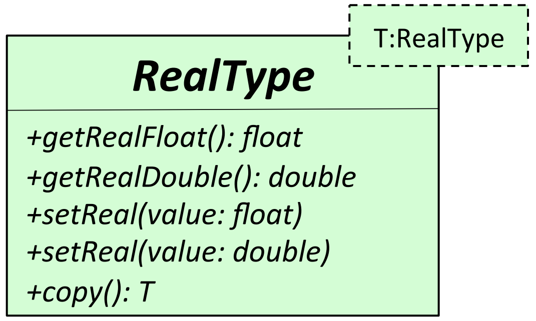
- Wrap any real valued image
- Use a localizing Cursor on the output image and a RealRandomAccess on the input to transform the image
- Each pixel location in the output needs to be transformed and the respective value read from the interpolated image
- ImgLib2_Transform.java


T:RealType
Img
+cursor(): Cursor<T>
+localizingCursor(): Cursor<T>
+randomAccess(): RandomAccess<T>
+firstElement(): T
+factory(): ImgFactory<T>
+numDimensions(): int
ImagePlus


T:RealType
(Localizing)Cursor
+hasNext()
+fwd()
+get(): T
+next(): T
+localize(position:int[])
+localize(position:long[])
+getIntPosition(d:int): int
+getLongPosition(d:int): long


T:RealType
RealRandomAccess
+get(): T
+fwd(d:int)
+bck(d:int)
+setPosition(position:float[])
+setPosition(position:double[])
+setPosition(pos:float, d:int)
+setPosition(pos:double, d:int)





ImgLib2

Special thanks to …

- <u>Tobias Pietzsch (MPI-CBG)</u>
- <u>Stephan Saalfeld (MPI-CBG)</u>

- Pavel Tomancak (MPI-CBG)
- Gene Myers (MPI-CBG)
- Rob Singer
  (Einstein College & Janelia Farm)

- ImageJ2 crew
  - Johannes Schindelin
  - Curtis Rueden
  - Barry DeZonia
  - Kevin Eliceiri
- Albert Cardona
- KNIME guys
  - Christian Dietz
  - Martin Horn

http://fiji.sc/~preibisch/ImgLib2_presentation.pdf

(this presentation as PDF)

http://fiji.sc/~preibisch/ImgLib2_Introduction_src.zip

(the sources for this workshop and completed examples)

http://fiji.sc/~preibisch/ImgLib2_Introduction.zip

(the sources for this workshop, completed examples and pictures)