```
ChangeLog - SimpleOpenNI
------------------------
http://code.google.com/p/simple-openni/

vx.xx status, mm/dd/yyyy

v0.xx , in planning:
                - *Remove all XML init. functions:
                - *Added methods for highperformance drawing of the depthMap
pointcloud (Opengl VBO)
                - *Multiple sensors with the generators
                - *Remove all deprecated functions, some callbacks will change
                - *Should make an example for calibration file
                - *Added support for Processing 2.0, still some open questions, but
it works roughly
                - *Added Kinect motor support(thanks to BitGriff):
                        - moveKinect(float angle) - tilts the kinect
                  Does not work with the newest kinect drivers(5.1.02,Linux64)

v2.00 , 12/01/2012,
        - OpenNI 2.0

v0.27 , 05/24/2012,
        - Use the new OpenNI modules:
          OpenNI v1.5.4.0
          NITE v1.5.2.21
          Sensor v5.1.2.1
          Kinect Sensor v0.92-v5.1.2.1
        - Added functions to setup a user defined coordinate system. This can be
helpfull in a lot of cases.
          For example if your camera is tilted but you want the camera depth data in
a coordinate system which
          is parallel to the floor. Another scenario could be that you have multiple
cameras, but you want
          all cameras in the same coordinate system.
          From the moment that you setup a user defined coordinate system, all 3d
related data from
          SimpleOpenNI will be transformed into the user defined coordinate system.
          New functions:
                boolean hasUserCoordsys()
                  - returns true if a user coordinate system was defined
                void resetUserCoordsys()
                  - removes the user coordinate system
                void setUserCoordsys(nullPoint3dX, nullPoint3dY, nullPoint3dZ,
                                                        xDirPoint3dX, xDirPoint3dY,
xDirPoint3dZ,
                                zDirPoint3dX, zDirPoint3dY, zDirPoint3dZ);
                  - sets the user defined coordinate system. The system if defined
by 3 realworld points.
                void getUserCoordsys(PMatrix3D mat)
                  - returns the orig.coordinatesystem to user.coordinatesystem
transformation matrix.
                void getUserCoordsysBack(PMatrix3D mat)
                  - returns the user.coordinatesystem to orig.coordinatesystem
transformation matrix.
```

```
        - Added a new example to show the user defined coordinate system:
          extensions/UserCoordsys
        - Added 2 new callback functions for the skeleton tracking:
          onExitUser(int userId)                -       informs when a user gets out
of view(onExitUser takes some time till it gives the call)
          onReEnterUser(int userId)             -       informs when a user reenters
the view
          Those callback are implemented in all examples which use skeletontracking
        - Added a new function to get the users for the skeleton tracking:
          int[] getUsers() - returns a list of all active users
        - Extended the User3d example. Shows how to calculate the direction of the
body based on the
          skeleton data. See the function 'getBodyDirection'.
        - Extended the skeleton tracking examples to show multiple tracked users
        - Added ray-triangle/sphere intersection function and an example source at
'extensions/IntersectionTest'
          static boolean rayTriangleIntersection(PVector p, PVector dir,

PVector vec0,PVector vec1, PVector vec2,

PVector hit)
          static int raySphereIntersection(PVector p, PVector dir,
                                                                        PVector
sphereCenter,float sphereRadius,
                                                                        PVector
hit1,PVector hit2)
        - Fixed the application export for win32/64 and linux32/64
        - Optimized the performance a little
        - Added Linux32 to the release

v0.26 , 02/29/2012,
                - Added the autocalibration, now you can only enter the scene and
get the skeleton data without the psi pose
                - Updated the examples to enable autocalibration:
                    - User
                    - User3d
                - Unified the SimpleOpenNI distribution library, from now there is
only one library distribution for OSX,Windows and Linux.
                  Had to make this change because of the Processing 2.0
autoinstaller.
                - SimpleOpenNI tries on windows/linux to find automatically the
valid architecture(32bit/64bit), depending on
                  java machine you use. On OSX this works already through the
universal libraries.
                  SimpleOpenNI will print out which version it uses.
                - Updated the wiki-install doc(thanks to Bradley Henke)


v0.25 , 01/03/2012,
                - Fixed the bug which prevented the valid playback of a recorded
file
                - Added new functions to control the playback if recorded files
                        - void setPlaybackSpeedPlayer(float speed)
                          sets the playback speed
                        - float playbackSpeedPlayer()
                          returns the playback speed
                        - void setRepeatPlayer(bool loop)
```

```
                        sets if the recorded file should be played in a
 loop(default: true)
                      - bool repeatPlayer()
                        returns the playing mode
                      - unsigned int curFramePlayer()
                        returns the current frame
                      - unsigned int framesPlayer()
                        returns the maximum frames for this record
                      - void seekPlayer(int offset,int seekType)
                        seeks to the offset
                        seekType:
                              PLAYER_SEEK_SET - jumps to the absolute position
                              PLAYER_SEEK_CUR - jumps relative, offset can be
 positive or negative
                              PLAYER_SEEK_END - jumps backwards, from the end,
 offset must be a negative value
                      - bool isEndPlayer()
                        returns if the player ended(works only if the repeatMode
 is set to false)
              - Updated and extended the RecordPlayer example. Now you have a
 timeline and you can control some playstates with the keys

 v0.24 , 12/27/2011,
              - Updated to versions:
                      OpenNI 1.5.2.7
                      NITE 1.5.2.7
                      SensorKinect 5.1.025
                      Sensor 5.1.025
              - Use new Avin2 Kinect drivers, now the mac starts up much faster!
              - Added better checks if SimpleOpenNI could be started. Extended all
 examples.
              - Added OpenNI install files for every release and plattform
              - Added support for export application(no applets), this application
 works on your machine, if you want to use the app on another machine,
                you still have to install OpenNI/NITE/Sensor libs.
                There is a bug in Processing 2.0 for linux, so the export won't
 work, only if you change the app script:
                      -Djava.library.path="$APPDIR:$APPDIR\lib"
                      to
                      -Djava.library.path="$APPDIR:$APPDIR/lib"
                http://code.google.com/p/processing/issues/detail?
 id=945&q=mrn&colspec=Stars%20ID%20Type%20Status%20Priority%     20Owner%20Summary
              - Updated install section on the wiki
              - Added classpaths to the JavaDoc generator

 v0.22 , 11/13/2011,
        New:
              - Updated the library with the new OpenNI/Nite versions. If you
 download there is a note which versions
                where used for which platform
              - Updated OSX installer
              - Added methods to access and query multiple cameras:
                Be aware that you shouldn't put the cameras at the same usb
 bus(usb performance!).
                - SimpleOpenNI(int cameraIndex,PApplet parent) - setup the
 specific camera
                - SimpleOpenNI.updateAll() - updates all cameras      (static
```

```
 method)
                  - int deviceCount() - returns  the count of cameras (static
 method)
                  - int deviceNames(StrVector strList) - gives a list of the cameras
 that are available and
                        returns the count (static method)
                - Added new example:
                  - 'MultiCam.pde' - Shows how to use multiple cameras
                  The multicamera support works at the moment only for the depthMap,
 rgbMap and the infraRedMap.
                  I experienced stange troubles if i use generators like
 SceneAnalyzer or UserGenerator. Seams to mix up
                  internaly the data. Have to find a fix for this behaviour.
                - Added methods to get callbacks from SimpleOpenNI to different
 classes than only PApplet
                  - enableHands(Object cbClass)
                  - enableGesture(Object cbClass)
                  - enableUser(int flags,Object cbClass) -
                - Added new example:
                  - 'User3dCallback.pde'
                - Added 64bit support for windows
                - Added a Sourcecode download, because the SVN is not always ready
 to compile

        Fix:
                - Updated new function names because of the function name changes of
 openNI + nite. Only internaly.

                - Had big troubles with the newest OpenNI 64bit version on linux
 11.10(Kubuntu). The depthMap was buggy(flashing) on
                  kinect + asus xtion pro. I had to change the OpenNI code +
 recompile the OpenNI library, see:
                  http://dancingwithyourshadow.blogspot.com/2011/10/using-kinect-
 with-openni-on-ubuntu-1110.html
                  The only thing i did was to replace the line 1057 in the file
 /Source/OpenNI/Linux/XnUSBLinux-x86.cpp
                  > memmove(pTransfer->buffer + nTotalBytes, pBuffer, pPacket-
 >actual_length);
                  Also i had troubles to get the kinect and the asus xtion to run
 both at the same time.
                  At the end it worked, i had to install the driver in a special
 order + use the right version:
                  - install 'sensor-bin-linux64-v5.0.4.3' ('sensor-bin-linux64-
 v5.0.4.4' didn't worked in combination with the kinect,
                        at the end the kinect worked but displayed only blank data)
                  - install 'SensorKinect-Bin-Linux64-v5.0.3.4'
                  - install again 'sensor-bin-linux64-v5.0.4.3'
                  In that way i managed that both sensors are working at the same
 time.

        Open Problems:
                - On OSX i didn't get any configuration to run the Asus Xtion Pro
 and the Kinect at the same time,
                  eather one of them worked or not. I tried with all current and old
 drivers

 v0.21 ,
```

```
        New:
                - Changed name of the InfraRed example 'IR' to
'DepthInfrared'.Processing had some troubles with the name.


v0.20 , 06/12/2011
        New:
                - Updated code to use OpenNI v1.1.041, Nite v1.3.1.5
                - Tested the library with Processing 1.5.1
                - Added 64bit support for Linux and OSX(32/64Bit), Windows will come
later, the Kinect driver is still only 32bit
                - Added a check to verify installed licenses, some error reports on
the list where caused, because the license
                  key was not proper installed, in this case there will be an error
message.
                - Added skeleton calibration file load/save:
                  see saveCalibrationDataSkeleton, loadCalibrationDataSkeleton
                  Also created an example for that:
examples/SimpleOpenNI/UserSaveCalib
                - Added AlternativeViewpoint capability, see:
examples/SimpleOpenNI/AlternativeViewPoint3d
                  The method 'alternativeViewPointDepthToImage()' aligns the Depth
data with the Image data
                - Added native OpenNI classes, like the ones for NITE, see
examples/SimpleOpenNI/UserSaveCalib, line 114
                  This is not fully functional at the moment(some nasty problems
with nested c++ classes), but in future i
                  would prefer to impelment those classes, that way you can use all
the features implemented in OpenNI, not
                  only the ones exported through the SimpleOpenNI class.
                  I also build in access methods in the SimpleOpenNI class,
                  this gives you the currently used generator(if you enabled it
before):
                        - getDepthGenerator()
                        - getImageGenerator()
                        - getIRGenerator()
                        - getSceneAnalyzer()
                        - getUserGenerator()
                        - getHandsGenerator()
                        - getGestureGenerator()
                  Those classes are implemented(no callbacks till now !):
                        - OutputMetaData
                        - MapMetaData
                        - DepthMetaData
                        - ImageMetaData
                        - IRMetaData
                        - SceneMetaData
                        - Capability
                        - SkeletonCapability
                        - PoseDetectionCapability
                        - MirrorCapability
                        - AlternativeViewPointCapability
                        - FrameSyncCapability
                        - NodeInfo
                        - NodeInfoList
                        - NodeWrapper
                        - ProductionNode
```

```
                        - Generator
                        - UserGenerator
                        - DepthGenerator
                        - ImageGenerator
                        - IRGenerator
                        - GestureGenerator
                        - HandsGenerator
                        - SceneAnalyzer
                - Added Nite extensions XnVPushDetector/XnVSwipeDetector from David
 Buchman. Had to change it a bit, due to changes
                  of Nite in the current version, this functions where removed:
                        SetSteadyMaxVelocity
                        GetSteadyMaxVelocity
                - Ported some more NITE classes, not very well tested, need to build
 examples:
                        - XnVSelectableSlider2D
                        - XnVSelectableSlider1D
                        - XnVWaveDetector
                        - XnVSteadyDetector
                        - XnVSlider1D
                        - XnVSlider2D
                        - XnVSlider3D
                - Added some new example:
                  examples/Nite/Slider2d        - Shows how to use 2D Sliders
                  examples/eclipse/                     - Shows how to use
 SimpleOpenNI in eclipse
                - Changed some Nite files, because of deprecated methods
                        -  XnVSessionManager: Initialize,SetGesture, SetQRGesture
                - Reorganized the sample folder structure and the names of the
 examples
                - Added build scripts for linux, see
 /trunk/SimpleOpenNI/buildLinux64.sh
                - Added FAQ, Screenshot section to the wiki
                - Updated install instructions on the wiki for linux
                - Fixed a bug: If you play a redord, skeletons were not displayed

 v0.18 alpha, 03/27/2011
        New:
                - Implemented multithreading support, the hole camera captureing
 will be made in an own thread.
                  By default it will still run in single threaded mode, to enable
 multithreading, just use this line:
                  context = new
 SimpleOpenNI(this,SimpleOpenNI.RUN_MODE_MULTI_THREADED);
                  So the drawing thread of processing is less stessed(the examples
 work now well, even on my Intel Atom/Linux).
                  Not all functions can profit that much from multithreading,
 'depthMapRealWorld' will still slow down the because
                  it copies w*h*Vector3d from c++ to java, which takes quite a
 will(30-40ms)
                  The hole implementation is not done, the callbacks of openNI are
 still from the other thread,
                  should be in the same thread as processing. For NITE this is
 already implemented.
                - Improved the performance in single threaded mode as well
                - Added new example:
                  SimpleOpenNI_DepthImage_Threaded - Show how to use multithreading
```

```
v0.17 alpha, 03/23/2011
        New:
                - Improved the overall performance
                - The skeleton was not always synchronized with the depthImage, now
this should be better


v0.16 alpha, 03/20/2011
        New:
                - Added NITE support, NITE is quite objectorientated so i couldn't
simplify that much, i just ported
                  most of the NITE classes to SimpleOpenNI. There are some NITE
classes that use multiple inheritance, which
                  java doesn't support. In this case i implemented the basic
functions that is needed(not all inheritated functions
                  are impemented)
                  NITE is not fully ported yet, with the next release it should be
done
                - Implemented a callback method from c++ to java, mostly for NITE
class callbacks. Turned out to be much more
                  work as expected, maybe there is a simpler way, but i didn't found
anything on the web.
                - Added new example:
                        SimpleOpenNI_NITE_Hands         - Shows how NITE works with
multiple hands(to activate multiple hands, see explanation in the sourcecode)
                        SimpleOpenNI_NITE_CircleCtrl- Shows circular control of NITE
                - Extended the java doc, now all classes will be visible with the
document generator
                - Fixed a typing mistake in the SimpleOpenNI_UserScene3d example

v0.15 alpha, 03/xx/2011
        New:
                - Added several functions for the skeleton handling + functions to
save and load the calibrations
                  for skeletons, but this saves and loads only to memory, not to the
disk, so it's just for one session.
                  bool saveCalibrationDataSkeleton(int user,int slot)   - saves the
calibration in ram
                  bool loadCalibrationDataSkeleton(int user,int slot)   - load the
calibration from ram
                  void clearCalibrationDataSkeleton(int slot)                   -
clears the calibration data at this slot
                  bool isCalibrationDataSkeleton(int slot)
- check if the calibration data exists at this slot
                  void abortCalibrationSkeleton(int user)
- aborts the calibration (requestCalibrationSkeleton)
                  bool isCalibratingSkeleton(int user)
- check if its in the state of calibrating
                  void setSmoothingSkeleton(float factor)
- set the smoothing factor for the skeleton

                  About save+load of calibration data, see further info on this
thread:
                  http://groups.google.com/group/openni-
dev/browse_thread/thread/f3def5f642802ec0/1bafdb1c8570bc70?
```

```
lnk=gst&q=Save++Calibration#1bafdb1c8570bc70
                - Added a download for linux-x86, also splitt up the download for
each version: Win/OSX/Linux
                - Extended the CMakeList.txt:
                        - updated the instructions to build, now all
builds(Win/OSX/Linux) works nearly the same
                        - now there should be a messages if you didn't setup the
environment variables(for linux)
                        - added the nite headers
                        - fixed a spelling bug, couldn't find smallcase openNI
library(for linux)
                - Added Eigen(http://eigen.tuxfamily.org) for internal use. Eigen is
a 3d Math Template Library.

v0.14 alpha, 03/11/2011
        New:
                - Added some functions for the scene analyser and user:
                int[]   sceneMap()                                              -
returns the hole scene map
                int[]   getUsersPixels(int user)        -       returns the sceneMap
for a user(only if user is enabled)
                int             getNumberOfUsers()                      -
returns current count of users
                boolean getCoM(int user,PVector com)-   returns the center of mass
for a user
                - Added new example:
                        SimpleOpenNI_UserScene3d
                                Shows how to get scene data for users
                - Fixed a bug in 'convertRealWorldToProjective'

v0.13 alpha, 03/06/2011
        New:
                - Created an installer for osx, but just for testing purpose
                - Removed the dependency from the xml init.file, added function, the
normal enable... functions
                  still work, they just use the default values:
                        init(PApplet parent)
                        enableDepth(int width,int height,int fps)
                        enableRGB(int width,int height,int fps)
                        enableIR(int width,int height,int fps)
                        enableScene(int width,int height,int fps)
                  Not all width/height/fps combinations are possible, depends on the
hardware!!!
                - Added record + save ability
                        enableRecorder(int recordMedium,String filePath)
                        addNodeToRecording(int nodeType,int compression)
                        removeNodeFromRecording(int nodeType)
                        openFileRecording(String filePath)
                  Still have to test how this works with the User/Hands/Gesture.
                - Added a function to get the orientation of a skeleton joint, the
return value is in a 3*3 Matrix
                  and the confidence of this joint
                        float getJointOrientationSkeleton(int userId,int
joint,PMatrix3D jointOrientation)
                  Updated SimpleOpenNI_User3d, now you also see the orientation of
the joints
                - Added new Examples:
```

```
                        SimpleOpenNI_RecorderPlay
                        SimpleOpenNI_DepthImageXml
        Changes:
                - All older examples where changed, now they are not using any xml
files

v0.12 alpha, 02/28/2011
        New:
                - Added handtracking
                  3 new callbacks:
                        onCreateHands(long nId, PVector pPosition, float fTime)
                        onUpdateHands(long nId, PVector pPosition, float fTime)
                        onDestroyHands(long nId, float fTime)
                - Added gesturetracking
                  2 new callbacks:
                        onRecognizeGesture(String strGesture, PVector idPosition,
PVector endPosition)
                        onProgressGesture(String strGesture, PVector position,float
progress)
                - Export now the realworld depthMap, see the function
depthMapRealWorld
                - Added functions to get the horizontal and the vertical field of
view of the camera
                  hFieldOfView
                  vFieldOfView
                - Added new Examples:
                        SimpleOpenNI_DepthMap3d
                        SimpleOpenNI_Hands3d
                - Added camera frustum drawing function 'drawCamFrustum'

        Changes:
                - Changed the names of the listed callbacks, just to have all over
the same naming method
                  onUserNew                     -> onNewUser
                  onUserLost                    -> onLostUser
                  onCalibrationStarted  -> onStartCalibration
                  onCalibrationEnded    -> onEndCalibration
                  onPoseStarted                 -> onStartPose
                  onPoseEnded                   -> onEndPose
                - Updated several examples


v0.11 alpha, 02/22/2011
        - started the changeLog
v0.10 alpha
        -
```