

a list of built-in objects in ofelia.

`declare -lib ofelia`

----- WINDOW -----	
<code>ofelia</code>	- initialize the ofelia external library
<code>ofWindow</code>	- handle the output window
<code>ofGetWidth</code>	- get the width of the current window
<code>ofGetHeight</code>	- get the height of the current window
<code>ofGetDimen</code>	- get the dimensions of the current window
<code>ofGetWindowScale</code>	- get the scale of the current window
<code>ofGetFrameNum</code>	- get the number of frames rendered
<code>ofGetFrameRate</code>	- get the actual frame rate of the current window
<code>ofGetTargetFrameRate</code>	- get the target frame rate of the current window
<code>ofGetElapsedTime</code>	- get the elapsed time in milliseconds
<code>ofGetLastFrameTime</code>	- get the last frame time in milliseconds
<code>ofGetOrienLock</code>	- get the orientation lock state of the current window
<code>ofGetOrien</code>	- get the orientation of the current window
<code>ofGetFullscreen</code>	- get the fullscreen state of the current window
<code>ofGetFocus</code>	- get the focus state of the current window
<code>ofGetWindowPosX</code>	- get the x position of the current window
<code>ofGetWindowPosY</code>	- get the y position of the current window
<code>ofGetWindowPos</code>	- get the position of the current window
<code>ofGetScreenWidth</code>	- get the width of the current device's screen
<code>ofGetScreenHeight</code>	- get the height of the current device's screen
<code>ofGetScreenDimen</code>	- get the dimensions of the current device's screen
<code>ofGetRetina</code>	- get the retina scale of the current device's screen
<code>ofGetBgColorR</code>	- get the r value of the background color
<code>ofGetBgColorG</code>	- get the g value of the background color
<code>ofGetBgColorB</code>	- get the b value of the background color
<code>ofGetBColor</code>	- get the background color of the current window
<code>ofGetWindow</code>	- check if a window exists
<code>ofGetFirstRenderOrder</code>	- get the first rendering order
<code>ofGetLastRenderOrder</code>	- get the last rendering order
<code>ofTouchListener</code>	- listen to the touch events
<code>ofMouseListener</code>	- listen to the mouse events
<code>ofScrollListener</code>	- listen to the mouse scroll events
<code>ofKeyListener</code>	- listen to the key events
<code>ofAccelerometer</code>	- listen to the accelerometer events
<code>ofWindowScaleListener</code>	- listen to the updated scale of the current window
<code>ofOrienListener</code>	- listen to the updated orientation of the current window
<code>ofFullscreenListener</code>	- listen to the fullscreen state of the current window
<code>ofFocusListener</code>	- listen to the focus state of the current window
<code>ofWindowPosListener</code>	- listen to the updated position of the current window
<code>ofWindowDestroyListener</code>	- listen to the creation/destruction of the current window
<code>ofWindowLoadBang</code>	- listen to the creation of the current window
<code>ofWindowCloseBang</code>	- listen to the destruction of the current window
<code>ofBackListener</code>	- listen to the back button press on android devices
----- GRAPHICS -----	
<code>ofHead</code>	- the start of a rendering chain
<code>ofTranslate</code>	- move along the coordinate system
<code>ofRotateX</code>	- rotate around the x-axis of the coordinate system
<code>ofRotateY</code>	- rotate around the y-axis of the coordinate system
<code>ofRotateZ</code>	- rotate around the z-axis of the coordinate system
<code>ofRotateXYZ</code>	- rotate around the xyz-axis of the coordinate system
<code>ofRotate</code>	- produce a rotation of angle around the vector
<code>ofScale</code>	- scale along the coordinate system
<code>ofPushMatrix</code>	- push the current matrix
<code>ofPopMatrix</code>	- pop the current matrix
<code>ofGetTranslate</code>	- get the current translate information
<code>ofGetRotate</code>	- get the current rotate information
<code>ofGetScale</code>	- get the current scale information
<code>ofSetColor</code>	- set the draw color
<code>ofSetBgColor</code>	- set the background color
<code>ofSetRectMode</code>	- set the align mode for drawing rectangular objects
<code>ofSetTextMode</code>	- set the align mode for drawing texts
<code>ofSetFillMode</code>	- set the fill mode for drawing shaped objects
<code>ofSetPolyMode</code>	- set the poly winding mode for drawing
<code>ofSetBlendMode</code>	- set the blend mode for drawing
<code>ofSetLineWidth</code>	- set the width of the lined objects
<code>ofSetLineSmoothing</code>	- enable/disable the smoothing for lines
<code>ofSetCircleRes</code>	- set the resolution for circular objects
<code>ofSetCurveRes</code>	- set the resolution for curved objects
<code>ofPushStyle</code>	- push the current style
<code>ofPopStyle</code>	- pop the current style
<code>ofPushMatrix</code>	- separate render chains in matrix
<code>ofSepStyle</code>	- separate render chains in style
<code>ofSeparator</code>	- separate render chains in matrix and style
<code>ofViewport</code>	- setup the drawing viewport
<code>ofSetDepthTest</code>	- enable/disable the depth test
<code>ofSetArbTex</code>	- enable/disable the use of ARB textures
<code>ofSetAntiAliasing</code>	- enable/disable the anti-aliasing for lines
<code>ofSetBgAuto</code>	- enable/disable the auto background clearing function
<code>ofClear</code>	- clear the color and depth bits of current renderer
<code>ofClearColor</code>	- clear the color bits of current renderer
<code>ofClearDepth</code>	- clear the depth bits of current renderer
<code>ofClearAlpha</code>	- clear the alpha channel of current renderer
<code>ofBeginShape</code>	- start drawing a new shape
<code>ofEndShape</code>	- finish drawing the shape and draw it to the screen
<code>ofNextContour</code>	- draw multiple contours within one shape
<code>ofVertex2d</code>	- specify a single 2d point of a shape
<code>ofVertex3d</code>	- specify a single 3d point of a shape
<code>ofCurveVertex2d</code>	- specify a single 2d point of a shape
<code>ofCurveVertex3d</code>	- specify a single 3d point of a shape
<code>ofBezierVertex2d</code>	- describe a bezier curve through three points of a shape
<code>ofBezierVertex3d</code>	- describe a bezier curve through three points of a shape
<code>ofCircle</code>	- draw a circle
<code>ofEllipse</code>	- draw an ellipse
<code>ofArc</code>	- draw an arc
<code>ofSector</code>	- draw a sector
<code>ofLine2d</code>	- draw a 2d line
<code>ofLine3d</code>	- draw a 3d line
<code>ofCurve2d</code>	- draw a 2d curve
<code>ofCurve3d</code>	- draw a 2d curve
<code>ofBezier2d</code>	- draw a 2d bezier curve
<code>ofBezier3d</code>	- draw a 3d bezier curve
<code>ofQuadBezier2d</code>	- draw a 2d quadratic bezier curve
<code>ofQuadBezier3d</code>	- draw a 3d quadratic bezier curve
<code>ofTriangle2d</code>	- draw a 2d triangle
<code>ofTriangle3d</code>	- draw a 3d triangle
<code>ofEqTriangle</code>	- draw an equilateral triangle
<code>ofIsoTriangle</code>	- draw an isosceles triangle
<code>ofQuad2d</code>	- draw a 2d quadrilateral
<code>ofQuad3d</code>	- draw a 3d quadrilateral
<code>ofSquare</code>	- draw a square
<code>ofRectangle</code>	- draw a rectangle
<code>ofRectRounded</code>	- draw a rounded rectangle with a given corner radius
<code>ofRectRounded4</code>	- draw a rounded rectangle with a given 4 corner radusses
<code>ofCross</code>	- draw a cross
<code>ofHeart</code>	- draw a heart
<code>ofMoon</code>	- draw a moon
<code>ofRegPolygon</code>	- draw a regular polygon
<code>ofStar</code>	- draw a star
<code>ofAxes</code>	- draw axes
<code>ofBox</code>	- draw a box
<code>ofCone</code>	- draw a cone
<code>ofCylinder</code>	- draw a cylinder
<code>ofIcosphere</code>	- draw an icosphere
<code>ofPlane</code>	- draw a plane
<code>ofSphere</code>	- draw a sphere
<code>ofArrow</code>	- draw an arrow
<code>ofGrid</code>	- draw grid planes
<code>ofGridPlane</code>	- draw a xy grid plane
<code>ofRotAxis</code>	- draw a set of 3-axis aligned circular bands
<code>ofLoadPolyline2d</code>	- store an array of polyline2d commands
<code>ofLoadPolyline3d</code>	- store an array of polyline3d commands
<code>ofDrawPolyline2d</code>	- draw the stored polyline2d
<code>ofDrawPolyline3d</code>	- draw the stored polyline3d
<code>ofDoesPolyline2dNameExist</code>	- check the existence of a polyline2d variable name
<code>ofDoesPolyline3dNameExist</code>	- check the existence of a polyline3d variable name
<code>ofEditPolyline2dPoint</code>	- edit the stored polyline2d point
<code>ofEditPolyline3dPoint</code>	- edit the stored polyline3d point
<code>ofGetPolyline2dPoint</code>	- get a polyline2d point at the given index
<code>ofGetPolyline3dPoint</code>	- get a polyline3d point at the given index
<code>ofGetPolyline2dPoints</code>	- get all polyline2d points as a list
<code>ofGetPolyline3dPoints</code>	- get all polyline3d points as a list
<code>ofIsPointInsidePolyline2d</code>	- check if a 2d point is within a closed polyline2d
<code>ofIsPointInsidePolyline3d</code>	- check if a 2d point is within a closed polyline3d
<code>ofGetPolyline2dCommand</code>	- get a polyline2d command at the given index
<code>ofGetPolyline3dCommand</code>	- get a polyline3d command at the given index
<code>ofGetPolyline2dCommands</code>	- get all polyline2d commands as a list
<code>ofGetPolyline3dCommands</code>	- get all polyline3d commands as a list
<code>ofGetPolyline2dBoundingBox</code>	- get the dimensions of the polyline2d bounding box
<code>ofGetPolyline3dBoundingBox</code>	- get the dimensions of the polyline3d bounding box
<code>ofGetPolyline2dCentroid</code>	- get the center position of the polyline2d area
<code>ofGetPolyline3dCentroid</code>	- get the center position of the polyline3d area
<code>ofGetPolyline2dArea</code>	- get the precise area of the polyline2d
<code>ofGetPolyline3dArea</code>	- get the precise area of the polyline3d
<code>ofGetPolyline2dPerimeter</code>	- get the size of the perimeter of the polyline2d
<code>ofGetPolyline3dPerimeter</code>	- get the size of the perimeter of the polyline3d
<code>ofLoadPath2d</code>	- store an array of path2d commands
<code>ofLoadPath3d</code>	- store an array of path3d commands
<code>ofDrawPath2d</code>	- draw the stored path2d
<code>ofDrawPath3d</code>	- draw the stored path3d
<code>ofDoesPath2dNameExist</code>	- check the existence of a path2d variable name
<code>ofDoesPath3dNameExist</code>	- check the existence of a path3d variable name
<code>ofGetPath2dPoint</code>	- get a path2d point at the given index
<code>ofGetPath3dPoint</code>	- get a path3d point at the given index
<code>ofGetPath2dPoints</code>	- get all path2d points as a list
<code>ofGetPath3dPoints</code>	- get all path3d points as a list
<code>ofIsPointInsidePath2d</code>	- check if a 2d point is within a closed path2d
<code>ofGetPath2dCommand</code>	- get a path2d command at the given index
<code>ofGetPath3dCommand</code>	- get a path3d command at the given index
<code>ofGetPath2dCommands</code>	- get all path2d commands as a list
<code>ofGetPath3dCommands</code>	- get all path3d commands as a list
<code>ofGetPath2dTessellation</code>	- get the tessellation data to convert path2d to mesh2d
<code>ofGetPath3dTessellation</code>	- get the tessellation data to convert path3d to mesh3d
<code>ofGetPath2dBoundingBox</code>	- get the dimensions of the path2d bounding box
<code>ofGetPath3dBoundingBox</code>	- get the dimensions of the path3d bounding box
<code>ofGetPath2dCentroid</code>	- get the center position of the path2d area
<code>ofGetPath3dCentroid</code>	- get the center position of the path3d area
<code>ofGetPath2dArea</code>	- get the precise area of the path2d
<code>ofGetPath3dArea</code>	- get the precise area of the path3d
<code>ofGetPath2dPerimeter</code>	- get the size of the perimeter of the path2d
<code>ofGetPath3dPerimeter</code>	- get the size of the perimeter of the path3d
<code>ofCreateFbo</code>	- create framebuffer object
<code>ofBindFboTex</code>	- bind the stored fbo's texture
<code>ofDrawFbo</code>	- draw the stored fbo
<code>ofDoesFboNameExist</code>	- check the existence of a fbo variable name
<code>ofIsFboAllocated</code>	- check if the fbo is allocated or not
<code>ofGetFboDimen</code>	- get the dimensions of the fbo
<code>ofGetFboType</code>	- get the type of the fbo
<code>ofGetFboMaxSamples</code>	- get the maximum number of MSAA samples
<code>ofCreateImage</code>	- create an image
<code>ofLoadImage</code>	- store an array of images
<code>ofEditImage</code>	- edit the stored image
<code>ofSaveImage</code>	- save image to disk
<code>ofBindImageTex</code>	- bind the stored image's texture
<code>ofDrawImage</code>	- draw the stored image
<code>ofDrawSubImage</code>	- draw a subsection of the image
<code>ofDoesImageNameExist</code>	- check the existence of an image variable name
<code>ofGetImagePath</code>	- get the absolute path of the image
<code>ofIsImageAllocated</code>	- check if the image is allocated or not
<code>ofGetImageDimen</code>	- get the dimensions of the image
<code>ofGetImageType</code>	- get the type of the image
<code>ofGetImageColorAt</code>	- get the color of a pixel at the specified x, y index
<code>ofGetImageTexCoord</code>	- get the texture coordinate of the image from 2d vertex
<code>ofGetImageTexCoords</code>	- get the texture coordinates of the image from 2d vertices
<code>ofLoadFont</code>	- store an array of fonts
<code>ofEditFont</code>	- edit the stored font
<code>ofBindFontTex</code>	- bind the stored font's texture
<code>ofDrawText</code>	- draw a text using the stored font
<code>ofDrawTextAsShapes</code>	- draw a text as shapes using the stored font
<code>ofDoesFontNameExist</code>	- check the existence of a font variable name
<code>ofGetFontPath</code>	- get the absolute path of the font
<code>ofGetFontSize</code>	- get the size of the font
<code>ofIsFontLoaded</code>	- check if the font is loaded or not
<code>ofGetTextBoundingBox</code>	- get the dimensions of the text bounding box
<code>ofGetFontLetterSpacing</code>	- get the letter spacing of the font
<code>ofGetFontLineHeight</code>	- get the line height of the font
<code>ofGetFontSpaceSize</code>	- get the space size of the font
<code>ofGetTextMesh2dCommands</code>	- get the mesh2d data based on the font and text
<code>ofGetTextMesh3dCommands</code>	- get the mesh3d data based on the font and text
<code>ofLoadMesh2d</code>	- store a set of arrays for a 2d mesh
<code>ofLoadMesh3d</code>	- store a set of arrays for a 3d mesh
<code>ofDrawMesh2d</code>	- draw the stored mesh2d
<code>ofDrawMesh3d</code>	- draw the stored mesh3d
<code>ofDoesMesh2dNameExist</code>	- check the existence of a mesh2d variable name
<code>ofDoesMesh3dNameExist</code>	- check the existence of a mesh3d variable name
<code>ofEditMesh2dVertex</code>	- edit the stored mesh2d vertex
<code>ofEditMesh3dVertex</code>	- edit the stored mesh3d vertex
<code>ofEditMesh2dIndex</code>	- edit the stored mesh2d index
<code>ofEditMesh3dIndex</code>	- edit the stored mesh3d index
<code>ofEditMesh2dNormal</code>	- edit the stored mesh2d normal
<code>ofEditMesh3dNormal</code>	- edit the stored mesh3d normal
<code>ofEditMesh2dTexCoord</code>	- edit the stored mesh2d texture coordinate
<code>ofEditMesh3dTexCoord</code>	- edit the stored mesh3d texture coordinate
<code>ofEditMesh2dColor</code>	- edit the stored mesh2d color
<code>ofEditMesh3dColor</code>	- edit the stored mesh3d color
<code>ofGetMesh2dVertex</code>	- get the mesh2d vertex at the given index
<code>ofGetMesh3dVertex</code>	- get the mesh3d vertex at the given index
<code>ofGetMesh2dIndex</code>	- get the mesh2d index at the given index
<code>ofGetMesh3dIndex</code>	- get the mesh3d index at the given index
<code>ofGetMesh2dNormal</code>	- get the mesh2d normal at the given index
<code>ofGetMesh3dNormal</code>	- get the mesh3d normal at the given index
<code>ofGetMesh2dTexCoord</code>	- get the mesh2d texture coordinate at the given index
<code>ofGetMesh3dTexCoord</code>	- get the mesh3d texture coordinate at the given index
<code>ofGetMesh2dColor</code>	- get the mesh2d color at the given index
<code>ofGetMesh3dColor</code>	- get the mesh3d color at the given index
<code>ofGetMesh2dVertices</code>	- get all mesh2d vertices as a list
<code>ofGetMesh3dVertices</code>	- get all mesh3d vertices as a list
<code>ofGetMesh2dIndices</code>	- get all mesh2d indices as a list
<code>ofGetMesh3dIndices</code>	- get all mesh3d indices as a list
<code>ofGetMesh2dNormals</code>	- get all mesh2d normals as a list
<code>ofGetMesh3dNormals</code>	- get all mesh3d normals as a list
<code>ofGetMesh2dTexCoords</code>	- get all mesh2d texture coordinates as a list
<code>ofGetMesh3dTexCoords</code>	- get all mesh3d texture coordinates as a list
<code>ofGetMesh2dColors</code>	- get all mesh2d colors as a list
<code>ofGetMesh3dColors</code>	- get all mesh3d colors as a list
<code>ofGetMesh2dCommands</code>	- get all mesh2d commands as a list
<code>ofGetMesh3dCommands</code>	- get all mesh3d commands as a list
<code>ofGetMesh2dBoundingBox</code>	- get the dimensions of the mesh2d bounding box
<code>ofGetMesh3dBoundingBox</code>	- get the dimensions of the mesh3d bounding box
<code>ofGetMesh2dCentroid</code>	- get the centroid of all the vetices in the mesh2d
<code>ofGetMesh3dCentroid</code>	- get the centroid of all the vetices in the mesh3d
<code>ofEasyCam</code>	- a simple camera for interacting with objects in 3d space
<code>ofCamera</code>	- a basic camera for interacting with objects in 3d space
<code>ofPointLight</code>	- a light that spreads outward evenly in all directions
<code>ofSpotLight</code>	- a light that spreads outward in a cone
<code>ofDirectionalLight</code>	- a light that comes evenly from a given direction
<code>ofMaterial</code>	- set the material of the object
----- TYPES -----	
<code>ofLoadFloat</code>	- store an array of floats
<code>ofEditFloat</code>	- edit the stored float
<code>ofDoesFloatNameExist</code>	- check the existence of a float variable name
<code>ofGetFloat</code>	- get a float element at the given index
<code>ofGetFloats</code>	- get all float elements as a list
<code>ofGetFloatAverage</code>	- get the average value of float elements
<code>ofLoadVec2f</code>	- store an array of two dimensional vectors
<code>ofEditVec2f</code>	- edit the stored vec2f
<code>ofDoesVec2fNameExist</code>	- check the existence of a vec2f variable name
<code>ofGetVec2f</code>	- get a vec2f element at the given index
<code>ofGetVec2fs</code>	- get all vec2f elements as a list
<code>ofGetVec2fAverage</code>	- get the average value of vec2f elements
<code>ofGetVec2fAngle</code>	- get the angle in degrees between two vec2fs
<code>ofGetVec2fAngleRad</code>	- get the angle in radians between two vec2fs
<code>ofGetVec2fDist</code>	- get the distance between two vec2fs
<code>ofGetVec2fDistSquared</code>	- get the squared distance between two vec2fs
<code>ofGetVec2fDot</code>	- get the dot product of two vec2fs
<code>ofGetVec2fLength</code>	- get the length of the vec2f element
<code>ofGetVec2fLengthSquared</code>	- get the squared length of the vec2f element
<code>ofLoadVec3f</code>	- store an array of three dimensional vectors
<code>ofEditVec3f</code>	- edit the stored vec3f
<code>ofDoesVec3fNameExist</code>	- check the existence of a vec3f variable name
<code>ofGetVec3f</code>	- get a vec3f element at the given index
<code>ofGetVec3fs</code>	- get all vec3f elements as a list
<code>ofGetVec3fAverage</code>	- get the average value of vec3f elements
<code>ofGetVec3fAngle</code>	- get the angle in degrees between two vec3fs
<code>ofGetVec3fAngleRad</code>	- get the angle in radians between two vec3fs
<code>ofGetVec3fDist</code>	- get the distance between two vec3fs
<code>ofGetVec3fDistSquared</code>	- get the squared distance between two vec3fs
<code>ofGetVec3fDot</code>	- get the dot product of two vec3fs
<code>ofGetVec3fLength</code>	- get the length of the vec3f element
<code>ofGetVec3fLengthSquared</code>	- get the squared length of the vec3f element
<code>ofLoadVec4f</code>	- store an array of four dimensional vectors
<code>ofEditVec4f</code>	- edit the stored vec4f
<code>ofDoesVec4fNameExist</code>	- check the existence of a vec4f variable name
<code>ofGetVec4f</code>	- get a vec4f element at the given index
<code>ofGetVec4fs</code>	- get all vec4f elements as a list
<code>ofGetVec4fAverage</code>	- get the average value of vec4f elements
<code>ofGetVec4fDist</code>	- get the distance between two vec4fs
<code>ofGetVec4fDistSquared</code>	- get the squared distance between two vec4fs
<code>ofGetVec4fDot</code>	- get the dot product of two vec4fs
<code>ofGetVec4fLength</code>	- get the length of the vec4f element
<code>ofGetVec4fLengthSquared</code>	- get the squared length of the vec4f element
<code>ofLoadColor</code>	- store an array of colors
<code>ofEditColor</code>	- edit the stored color
<code>ofDoesColorNameExist</code>	- check the existence of a color variable name
<code>ofGetColor</code>	- get a color element at the given index
<code>ofGetColors</code>	- get all color elements as a list
<code>ofLoadSymbol</code>	- store an array of symbols
<code>ofEditSymbol</code>	- edit the stored symbol
<code>ofDoesSymbolNameExist</code>	- check the existence of a symbol variable name
<code>ofGetSymbol</code>	- get a symbol element at the given index
<code>ofGetSymbols</code>	- get all symbol elements as a list
----- MATH -----	
<code>ofAngleDifferenceDegrees</code>	- calculate the difference between two angles in degrees
<code>ofAngleDifferenceRadians</code>	- calculate the difference between two angles in radians
<code>ofDegToRad</code>	- convert degrees to radians
<code>ofRadToDeg</code>	- convert radians to degrees
<code>ofDist2d</code>	- calculate the 2d distance between two points
<code>ofDist3d</code>	- calculate the 3d distance between two points
<code>ofDistSquared2d</code>	- calculate the squared 2d distance between two points
<code>ofDistSquared3d</code>	- calculate the squared 3d distance between two points
<code>ofInRange</code>	- determine if a number is inside of a given range
<code>ofClamp</code>	- clamp a value between min and max
<code>ofNormalize</code>	- map the input value to be within 0 and 1
<code>ofLerp</code>	- linearly interpolate a value within a range
<code>ofLerpDegrees</code>	- linearly interpolate a value between two angles in degrees
<code>ofLerpRadians</code>	- linearly interpolate a value between two angles in radians
<code>ofRandom</code>	- get a random number within a given range
<code>ofRandomf</code>	- get a random floating point number between -1 and 1
<code>ofRandomur</code>	- get a random floating point number between 0 and 1
<code>ofSeedRandom</code>	- seed the random number generator with a unique value
<code>ofWrap</code>	- wrap a value if it overflows a given range
<code>ofWrapDegrees</code>	- wrap a value within the angle in degrees
<code>ofWrapRadians</code>	- wrap a value within the angle in radians
<code>ofMap</code>	- map the value to a new value
<code>ofNextPow2</code>	- calculate the next larger power of 2
<code>ofNoise</code>	- calculate a simplex noise value between 0 and 1
<code>ofSignedNoise</code>	- calculate a simplex noise value between -1 and 1
<code>ofSign</code>	- get the sign of a value
----- UTILS -----	
<code>ofAppend</code>	- append a symbol to an incoming message
<code>ofPrepend</code>	- prepend a symbol to an incoming message
<code>ofPack</code>	- combine several atoms into one message
<code>ofListFind</code>	- get indices of sublists found in a list
<code>ofFindList</code>	- get indices of sublists found in a list
<code>ofListInsert</code>	- insert a list into a list
<code>ofInsertList</code>	- insert a list into a list
<code>ofListFill</code>	- fill a list with element
<code>ofFillList</code>	- fill a list with element
<code>ofListReplace</code>	- replace sublists in a list
<code>ofReplaceList</code>	- replace sublists in a list
<code>ofListRemove</code>	- remove sublists in a list
<code>ofRemoveList</code>	- remove sublists in a list
<code>ofListErase</code>	- remove a range of elements from a list
<code>ofEraseList</code>	- remove a range of elements from a list
<code>ofListSort</code>	- sort a list in ascending or descending order
<code>ofListUnique</code>	- remove duplicates from a list
<code>ofListReverse</code>	- reverse the order of a list
<code>ofListShuffle</code>	- randomly change the order of a list
<code>ofListToSymbol</code>	- convert a list into a symbol
<code>ofSymbolToList</code>	- convert a symbol into a list
<code>ofHexToHsb</code>	- convert hex color values to hsb color values
<code>ofHexToRgb</code>	- convert hex color values to rgb color values
<code>ofHsbToHex</code>	- convert hsb color values to hex color values
<code>ofHsbToRgb</code>	- convert hsb color values to rgb color values
<code>ofRgbToHex</code>	- convert rgb color values to hex color values
<code>ofRgbToHsb</code>	- convert rgb color values to hsb color values
<code>ofValue</code>	- nonlocal shared value
<code>ofSend</code>	- send messages without patch cords
<code>ofReceive</code>	- receive messages without patch cords
<code>ofExpr</code>	- expression evaluation object
<code>ofDefine</code>	- expression evaluation object
<code>ofPatch</code>	- open/close pd patches
<code>ofGetCanvasName</code>	- get the unique name of the canvas
<code>ofGetDollarZero</code>	- get the \$0 value of the patch
<code>ofGetDolLarArgs</code>	- get the arguments of the patch
<code>ofError</code>	- print an error to the pd console
<code>ofFile</code>	- create/remove/rename/copy/move files
<code>ofSaveURL</code>	- save a file from a url
<code>ofDoesFileExist</code>	- check the existence of a file
<code>ofGetDirectoryFileNames</code>	- get the list of file names in a directory
<code>ofGetDirectoryFilePaths</code>	- get the list of file paths in a directory
<code>ofDirectory</code>	- create/remove/rename/copy/move directories
<code>ofGetPatchDirectory</code>	- get the directory of the patch
<code>ofGetHomeDirectory</code>	- get the user home directory on desktop platforms
<code>ofGetDocumentsDirectory</code>	- get the documents directory on ios devices
<code>ofGetLibraryDirectory</code>	- get the library directory on ios devices
<code>ofGetTemporaryDirectory</code>	- get the temporary directory on ios devices
<code>ofDoesDirectoryExist</code>	- check the existence of a directory
<code>ofGetAudioDevices</code>	- get a list of all available input/output audio devices
<code>ofSetAudioDevices</code>	- set input/output audio device, sample rate and block size
<code>ofGetMidiDevices</code>	- get a list of all available input/output midi devices
<code>ofSetMidiDevices</code>	- set input/output midi device
<code>ofCount</code>	- count over a range
<code>ofCountUntil</code>	- count over a range at once
<code>ofStep</code>	- increase or decrease a value in steps
<code>ofAnimate</code>	- smoothly change a value over time
<code>ofWith</code>	- pass messages from an explicit inlet
<code>ofGate</code>	- route a message to a output
<code>ofGetMinFloat</code>	