ActionScript 3.0 Class Diagram

Welcome to ActionScript 3.0, the new ECMA based language from Adobe Systems. This diagram shows all intrinsic classes, methods, and properties available within Adobe Flash Player 9. **Better by Adobe**.

Legend

- Superclass
- «interface» Interface — property: dataType; method(): returnType
- Superclass package::Class — property: dataType; method(): returnType
- implements

Error classes

Object **Error** — message: String; name: String; Error(String): Object; getStackTrace(): String; toString(): String

- ArgumentError
- DefinitionError
- EvalError
- RangeError
- ReferenceError
- SecurityError
- SyntaxError
- TypeError
- URIError
- VerifyError
- flash.errors::StackOverflowError
- flash.errors::ScriptTimeoutError
- flash.errors::MemoryError
- flash.errors::InvalidSWFError
- flash.errors::IllegalOperationError
- flash.errors::IOError
- flash.errors::EOFError

flash.media

EventDispatcher **Camera** — activityLevel: Number; bandwidth: int; currentFPS: Number; fps: Number; height: int; index: int; motionLevel: int; motionTimeout: int; muted: Boolean; name: String; names: Array; quality: int; width: int; getCamera(): Camera; setMode(); setMotionLevel(); setQuality()

EventDispatcher **Sound** — bytesLoaded: uint; bytesTotal: int; id3: ID3Info; isBuffering: Boolean; length: Number; url: String; close(); load(); play(): SoundChannel; Sound(): Object

EventDispatcher **SoundChannel** — leftPeak: Number; position: Number; rightPeak: Number; soundTransform : SoundTransform; stop()

Object **SoundLoaderContext** — bufferTime: Number; checkPolicyFile: Boolean; SoundLoaderContext(): Object

Object **SoundTransform** — leftToLeft: Number; leftToRight: Number; pan: Number; rightToLeft: Number; rightToRight: Number; volume: Number; SoundTransform(): Object

Object **ID3Info** — album: String; artist: String; comment: String; genre: String; songName: String; track: String; year: String

DisplayObject **Video** — deblocking: int; smoothing: Boolean; videoHeight: int; videoWidth: int; attachCamera(); attachNetStream () ; clear(); Video(): Object

EventDispatcher **Microphone** — activityLevel: Number; gain: Number; index: int; muted: Boolean; name: String; names: Array; rate: int; silenceLevel: Number; silenceTimeout: int; soundTransform : SoundTransform; useEchoSuppression: Boolean; getMicrophone(): Microphone; setLoopBack(); setSilenceLevel(); setUseEchoSuppression()

Object **SoundMixer** — bufferTime: int; soundTransform: SoundTransform; areSoundsInaccessible(): Boolean; computeSpectrum(); stopAll()

flash.geom

Object **ColorTransform** — alphaMultiplier: Number; alphaOffset: Number; blueMultiplier: Number; blueOffset: Number; color: uint; greenMultiplier: Number; greenOffset: Number; redMultiplier: Number; redOffset: Number; ColorTransform(): Object; concat(); toString(): String

Object **Point** — length: Number; x: Number; y: Number; add(): Point; clone(): Point; distance(): Number; equals(): Boolean; interpolate(): Point; normalize(); offset(); Point(): Object; polar(): Point; subtract(): Point; toString(): String

Object **Matrix** — a: Number; b: Number; c : Number; d: Number; tx: Number; ty: Number; clone(): Matrix; concat(); createBox(); createGradientBox(); deltaTransformPoint(): Point; identity(); invert(); Matrix(): Object; rotate(); scale(); toString(): String; transformPoint(): Point; translate()

Object **Rectangle** — bottom : Number; bottomRight: Point; height: Number; left: Number; right: Number; size: Point; top: Number; topLeft: Point; width: Number; x: Number; y: Number; clone(): Rectangle; contains(): Boolean; containsPoint(): Boolean; containsRect(): Boolean; equals(): Boolean; inflate(); inflatePoint(); intersection(): Rectangle; intersects(): Boolean; isEmpty(): Boolean; offset(); offsetPoint(); Rectangle(): Object; setEmpty(); toString(): String; union(): Rectangle

Object **Transform** — colorTransform: ColorTransform; concatenatedColorTransform: ColorTransform; concatenatedMatrix: Matrix; matrix: Matrix; pixelBounds: Rectangle

flash.accessibility

Object **Accessibility** — active: Boolean; updateProperties()

Object **AccessibilityProperties** — description: String; forceSimple: Boolean; name: String; noAutoLabeling: Boolean; shortcut: String; silent: Boolean; AccessibilityProperties(): Object

flash.display

EventDispatcher **DisplayObject** — accessibilityProperties: AccessibilityProperties; alpha: Number; blendMode: String; c acheAsBitmap: Boolean; filters: Array; height: Number; loaderInfo: LoaderInfo; mask: DisplayObject; mouseX: Number; mouseY: Number; name: String; opaqueBackground: Object; parent: DisplayObjectContainer; root: DisplayObject; rotation: Number; scaleRGrid: Rectangle; scaleX: Number; scaleY: Number; scrollRect: Rectangle; stage: Stage; transform: Transform; visible: Boolean; width: Number; x: Number; y: Number; getBounds(): Rectangle; getRect(): Rectangle; globalToLocal(): Point; hitTestObject(): Boolean; hitTestPoint(): Boolean; localToGlobal(): Point

«interface» **IBitmapDrawable**

Bitmap — bitmapData: BitmapData; pixelSnapping: String; smoothing: Boolean; Bitmap(): Object

AVM1Movie

MorphShape

Shape — graphics: Graphics; Shape(): Object

InteractiveObject — contextMenu: ContextMenu; doubleClickEnabled: Boolean; focusRect: Object; mouseEnabled: Boolean; tabEnabled: Boolean; tabIndex: int; InteractiveObject(): Object

Object **BitmapData** — height: int; rect: Rectangle; transparent: Boolean; width: int; applyFilter(); BitmapData(): Object; clone(): BitmapData; colorTransform () ; com pare(): Object; copyChannel(); copyPixels(); dispose(); draw(); fillRec t(); floodFill(); generateFilterRect(): Rectangle; getColorBoundsRect(): Rectangle; getPixel(): uint; getPixel32(): uint; getPixels(): ByteArray; hitTest(): Boolean; lock(); merge(); noise(); paletteMap(); perlinNoise(); pixelDissolve(): int; scroll(); setPixel(); setPixel32(); setPixels(); threshold(): uint; unlock()

Object **FrameLabel** — frame: int; name: String

Object **Scene** — labels: Array; name: String; numFrames: int

Object **ActionScriptVersion** — ACTIONSCRIPT2: uint; ACTIONSCRIPT3: uint

Object **BitmapDataChannel** — ALPHA : uint; BLUE: uint; GREEN: uint; RED: uint

Object **BlendMode** — ADD: String; ALPHA : String; DARKEN: String; DIFFERENCE: String; ERASE: String; HARDLIGHT: String; INVERT: String; LAYER: String; LIGHTEN: String; MULTIPLY: String; NORMA L: String; OVERLAY: String; SCREEN: String; SUBTRACT: String

Object **CapStyle** — NONE: String; ROUND: String; SQUARE: String

Object **GradientType** — LINEAR: String; RADIAL: String

Object **InterpolationMethod** — LINEAR_RGB: String; RGB: String

Object **JointStyle** — BEVEL: String; MITER: String; ROUND: String

Object **LineScaleMode** — HORIZONTAL: String; NONE: String; NORMAL: String; VERTICAL: String

Object **PixelSnapping** — ALWAYS: String; AUTO: String; NEVER: String

Object **SpreadMethod** — PAD: String; REFLECT: String; REPEAT: String

Object **StageAlign** — BOTTOM: String; BOTTOM_LEFT: String; BOTTOM_RIGHT: String; LEFT: String; RIGHT: String; TOP: String; TOP_LEFT: String; TOP_RIGHT: String

Object **StageQuality** — BEST: String; HIGH: String; LOW: String; MEDIUM: String

Object **StageScaleMode** — EXACT_FIT: String; NO _BORDER: String; NO _SCALE: String; SHOW _ALL: String

Object **SWFVersion** — FLASH1: uint; FLASH2: uint; FLASH3: uint; FLASH4: uint; FLASH5: uint; FLASH6: uint; FLASH7: uint; FLASH8: uint; FLASH9_5: uint

SimpleButton — downState: DisplayObject; enabled: Boolean; hitTestState: DisplayObject; overState: DisplayObject; soundTransform: SoundTransform; trackAsMenu: Boolean; upState: DisplayObject; useHandCursor: Boolean; SimpleButton(): Object

DisplayObjectContainer — mouseChildren: Boolean; num Children: int; tabChildren: Boolean; textSnapshot: TextSnapshot; addChild(): DisplayObject; addChildAt(): DisplayObject; areInaccessibleObjectsUnderPoint(): Boolean; contains(): Boolean; DisplayObjectContainer(): Object; getChildAt(): DisplayObject; getChildByName(): DisplayObject; getChildIndex(): int; getObjectsUnderPoint(): Array; removeChild(): DisplayObject; removeChildA t(): DisplayObject; setChildIndex(); swapChildren(); swapChildrenAt()

Loader — content: DisplayObject; contentLoaderInfo: LoaderInfo; close(); load(); loadBytes(); Loader(): Object; unload()

EventDispatcher **LoaderInfo** — actionScriptVersion: uint; applicationDomain: ApplicationDomain; bytesLoaded: uint; bytesTotal: uint; childAllowsParent: Boolean; content: DisplayObject; contentType: String; frameRate: Number; height: int; loader: Loader; loaderURL: String; parameters: Object; parentAllowsChild: Boolean; sameDomain: Boolean; sharedEvents: EventDispatcher; swfVersion: uint; url: String; width: int

Stage — align: String; focus: InteractiveObject; frameRate: Number; quality: String; scaleMode: String; showDefaultContextMenu: Boolean; stageFocusRect: Boolean; stageHeight: int; stageWidth: int; invalidate(); isFocusInaccessible(): Boolean

Object **Graphics** — beginBitmapFill(); beginFill(); beginGradientFill(); clear() ; curveTo(); drawCircle(); drawEllipse(); drawRec t(); drawRoundRect(); endFill(); lineGradientStyle() lineStyle(); lineTo() ; moveTo()

Sprite — buttonMode: Boolean; dropTarget: DisplayObject; graphics: Graphics; hitArea: Sprite; soundTransform: SoundTransform; useHandCursor: Boolean; Sprite(): Object; startDrag(); stopDrag()

MovieClip — currentFrame: int; currentLabel: String; currentLabels: A rray; currentScene: Scene; enabled: Boolean; framesLoaded: int; scenes: Array; totalFrames: int; trackAsMenu: Boolean; gotoAndPlay(); gotoAndStop(); MovieClip(): Object; nextFrame(); nextScene() ; play() ; prevFrame() ; prevScene() ; stop()

flash.events

EventDispatcher **ActivityEvent** — activating: Boolean; ACTIVITY: String; ActivityEvent(): Object; clone(): Event; toString(): String

EventDispatcher **ContextMenuEvent** — contextMenuOwner: InteractiveObject; MENU_ITEM_SELECT: String; MENU_SELECT: String; mouseTarget: InteractiveObject; clone(): Event; ContextMenuEvent(): Object; toString(): String

EventDispatcher **FocusEvent** — FOCUS_IN: String; FOCUS_OUT: String; KEY_FOCUS_CHANGE: String; keyCode: uint; MOUSE_FOCUS_CHANGE: String; relatedObject: InteractiveObject; shiftKey: Boolean; clone(): Event; FocusEvent(): Object; toString(): String

EventDispatcher **HTTPStatusEvent** — HTTP_STATUS: String; status: int; clone(): Event; HTTPStatusEvent(): Object; toString(): String

EventDispatcher **KeyboardEvent** — altKey: Boolean; charCode: uint; ctrlKey: Boolean; KEY_DOWN: String; KEY_UP: String; keyCode: uint; keyLocation: uint; shiftKey: Boolean; clone(): Event; KeyboardEvent(): Object; toString(): String; updateAfterEvent()

«interface» **IEventDispatcher** — addEventListener() ; dispatchEvent(): Boolean; hasEventListener() : Boolean; removeEventListener() ; willTrigger(): Boolean

implements

Object **EventDispatcher** — addEventListener() ; dispatchEvent(): Boolean; EventDispatcher(): Object; hasEventListener(): Boolean; removeEventLisitener() ; willTrigger(): Boolean

Object **Event** — ACTIVATE: String; ADDED: String; bubbles: Boolean; CANCEL: String; cancelable: Boolean; CHANGE: String; CLOSE: String; COMPLETE: String; CONNECT: String; currentTarget: Object; DEACTIVATE: String; ENTER_FRAME: String; eventPhase: uint; ID3: String; INIT: String; MOUSE_LEA VE: String; OPEN: String; REMOVED: String; RENDER: String; RESIZE: String; SCROLL: String; SELECT: String; SOUND_COMPLETE: String; TAB_CHILDREN_CHANGE: String; TAB_ENABLED_CHANGE: String; TAB_INDEX_CHANGE: String; target: Object; type: String; UNLOAD: String; clone(): Event; Event(): Object; formatToString(): String; isDefaultPrevented(): Boolean; preventDefault() ; stopImmediatePropagation() ; stopPropagation() ; toString(): String

EventDispatcher **TimerEvent** — TIMER: String; TIMER_COMPLETE: String; clone(): Event; TimerEvent(): Object; toString(): String; updateAfterEvent()

EventDispatcher **SyncEvent** — changeList: Array; SYNC: String; clone(): Event; SyncEvent(): Object; toString(): String

EventDispatcher **StatusEvent** — code: String; level: String; STATUS: String; clone(): Event; StatusEvent(): Object; toString(): String

EventDispatcher **ProgressEvent** — bytesLoaded: uint; bytesTotal: uint; PROGRESS: String; SOCKET_DATA : String; clone(): Event; ProgressEvent(): Object; toString(): String

EventDispatcher **NetStatusEvent** — info: Object; NET_STATUS: String; clone(): Event; NetStatusEvent(): Object; toString(): String

EventDispatcher **MouseEvent** — altKey: Boolean; buttonDown: Boolean; CLICK: String; ctrlKey: Boolean; delta: int; DOUBLE_CLICK: String; localX: Number; localY: Number; MOUSE_DOWN: String; MOUSE_MOVE: String; MOUSE_OUT: String; MOUSE_OVER: String; MOUSE_UP: String; MOUSE_WHEEL: String; relatedObject: InteractiveObject; ROLL_OUT: String; ROLL_OVER: String; shiftKey: Boolean; stageX: Number; stageY: Number; clone(): Event; MouseEvent(): Object; toString(): String; updateAfterEvent()

EventDispatcher **TextEvent** — LINK: String; text: String; TEXT_INPUT: String; clone(): Event; TextEvent(): Object; toString(): String

EventDispatcher **DataEvent** — data: String; DATA : String; clone(): Event; DataEvent(): Object; toString(): String

EventDispatcher **ErrorEvent** — ERROR: String; clone(): Event; ErrorEvent(): Object; toString(): String

EventDispatcher **IMEEvent** — IME_COMPOSITION: String; clone(): Event; IMEEvent(): Object; toString(): String

EventDispatcher **IOErrorEvent** — IO_ERROR: String; clone(): Event; IOErrorEvent(): Object; toString(): String

EventDispatcher **SecurityErrorEvent** — SECURITY_ERROR: String; clone(): Event; SecurityErrorEvent(): Object; toString(): String

Object **EventPhase** — AT_TARGET: uint; BUBBLING_PHASE: uint; CAPTURING_PHASE: uint

flash.filters

Object **BitmapFilterType** — FULL: String; INNER: String; OUTER: String

Object **BitmapFilterQuality** — HIGH: int; LOW: int; MEDIUM: int

BevelFilter — angle: Number; blurX: Number; blurY: Number; distance: Number; highlightAlpha: Number; highlightColor: uint; knockout: Boolean; quality: int; shadowAlpha: Number; shadowColor: uint; strength: Number; type: String; BevelFilter(): Object; clone(): BitmapFilter

ConvolutionFilter — alpha: Number; bias: Number; clamp: Boolean; color: uint; divisor: Number; matrix: Array; matrixX: Number; matrixY: Number; preserveAlpha: Boolean; clone(): BitmapFilter; ConvolutionFilter(): Object

GradientGlowFilter — alphas: Array; angle: Number; blurX: Number; blurY: Number; c olors: Array; distance: Number; knockout: Boolean; quality: int; ratios: Array; strength: Number; type: String; clone(): BitmapFilter; GradientGlowFilter(): Object

DropShadowFilter — alpha: Number; angle: Number; blurX: Number; blurY: Number; color: uint; distance: Number; hideObject: Boolean; inner: Boolean; knockout: Boolean; quality: int; strength: Number; clone(): BitmapFilter; DropShadowFilter(): Object

GradientBevelFilter — alphas: Array; angle: Number; blurX: Number; blurY: Number; colors: Array; distance: Number; knockout: Boolean; quality: int; ratios: Array; strength: Number; type: String; clone(): BitmapFilter; GradientBevelFilter(): Object

Object **BitmapFilter** — clone(): BitmapFilter

ConvolutionFilter — alpha: Number; color: uint; componentX: uint; componentY: uint; mapBitmap: BitmapData; mapPoint: Point; mode: String; scaleX: Number; scaleY: Number; clone(): BitmapFilter; DisplacementMapFilter(): Object

GlowFilter — alpha: Number; blurX: Number; blurY: Number; color: uint; inner: Boolean; knockout: Boolean; quality: int; strength: Number; clone(): BitmapFilter; GlowFilter(): Object

ColorMatrixFilter — matrix: Array; clone(): BitmapFilter; ColorMatrixFilter(): Object

BlurFilter — blurX: Number; blurY: Number; quality: int; BlurFilter(): Object; clone(): BitmapFilter

Object **DisplacementMapFilterMode** — CLAMP: String; COLOR: String; IGNORE: String; WRAP: String

flash.utils

«interface» **IDataInput** — bytesAvailable: uint; endian: String; objectEncoding: uint; readBoolean(): Boolean; readByte(): int; readBytes() ; readDouble(): Number; readFloat(): Number; readInt(): int; readMultiByte(): String; readObject(): Object; readShort(): int; readUnsignedByte(): uint; readUnsignedInt(): uint; readUnsignedShort(): uint; readUTF(): String; readUTFBytes(): String

implements

Object **ByteArray** — bytesAvailable: uint; defaultObjectEncoding: uint; endian: String; length: uint; objectEncoding: uint; position: uint; ByteArray(): Object; compress() ; readBoolean(): Boolean; readByte(): int; readBytes() ; readDouble(): Number; readFloat(): Number; readInt(): int; readMultiByte(): String; readObject(): Object; readShort(): int; readUnsignedByte(): uint; readUnsignedInt(): uint; readUnsignedShort(): uint; readUTF(): String; readUTFBytes(): String; uncompress() ; writeBoolean() ; writeByte() ; writeBytes() ; writeDouble() ; writeFloat() ; writeInt() ; writeMultiByte() ; writeObject() ; writeShort() ; writeUnsignedInt() ; writeUnsigned Int() ; writeUTF() ; writeUTFBytes()

implements

«interface» **IDataOutput** — endian: String; objectEncoding: uint; writeBoolean() ; writeByte() ; writeBytes() ; writeDouble() ; writeFloat() ; writeInt() ; writeMultiByte() ; writeObject() ; writeShort() ; writeUnsigned Int() ; writeUnsigned Int() ; writeUTF() ; writeUTFBytes()

EventDispatcher **Timer** — reset() ; start() ; stop() ; Timer() : Object

Object **Dictionary**

Object **Endian** — BIG_ENDIAN: String; LITTLE_ENDIAN: String

Object **Proxy** — callProperty(): Object; deleteProperty(): Boolean; getDescendants(): Object; getProperty(): Object; hasProperty(): Boolean; isAttribute(): Boolean; nextNameI ndex(): int; nextValue(): Object; setProperty()

«interface» **IExternalizable** — readExternal() ; writeExternal()

flash.external

Object **ExternalInterface** — available: Boolean; objectID: String; addCallback() ; call() : Object

flash.ui

Object **Keyboard** — BACKSPACE: uint; CAPS _LOCK: uint; capsLock: Boolean; CONTROL: uint; DELETE: uint; DOWN: uint; END: uint; ENTER: uint; ESCAPE: uint; F1: uint; F10: uint; F11: uint; F12: uint; F13: uint; F14: uint; F15: uint; F2: uint; F3: uint; F4: uint; F5: uint; F6: uint; F7: uint; F8: uint; F9: uint; HOME: uint; INSERT: uint; LEFT: uint; numLock: Boolean; NUMPAD_0: uint; NUMPAD_1: uint; NUMPAD_2: uint; NUMPAD_3: uint; UMPAD_4: uint; NUMPAD_5: uint; NUMPAD_6: uint; NUMPAD_7: uint; NUMPA D_8: uint; NUMPAD_9: uint; NUMPAD_ADD: uint; NUMPAD_DECIMAL: uint; NUMPAD_DIVIDE: uint; NUMPAD_ENTER: uint; NUMPAD_MULTIPLY: uint; NUMPAD_SUBTRACT: uint; PAGE_DOWN: uint; PAGE_UP: uint; RIGHT: uint; SHIFT: uint; SPACE: uint; TAB: uint; UP: uint; isAccessible(): Boolean

EventDispatcher **ContextMenu** — builtInItems: ContextMenuBuiltinItems; customItems: Array; clone(): ContextMenu; ContextMenu(): Object; hideBuiltInItems()

EventDispatcher **ContextMenuItem** — caption: String; enabled: Boolean; separatorBefore: Boolean; visible: Boolean; clone(): ContextMenuItem; ContextMenuItem(): Object

Object **ContextMenuBuiltInItems** — forwardAndBack: Boolean; loop: Boolean; play: Boolean; print: Boolean; quality: Boolean; rewind: Boolean; save: Boolean; zoom: Boolean; ContextMenuBuiltInItems(): Object

Object **KeyLocation** — LEFT: uint; NUM_PAD: uint; RIGHT: uint; STANDARD: uint

Object **Mouse** — hide() ; show()t

flash.text

«interface» **TextField** — alwaysShowSelection: Boolean; antiAliasType: String; autoSize: String; background: Boolean; backgroundColor: uint; border: Boolean; borderColor: uint; bottomScrollV: int; caretIndex: int; condenseWhite: Boolean; defaultTextFormat: TextFormat; displayAsPassword: Boolean; embedFonts: Boolean; gridFitType: String; htmlText: String; length: int; maxChars: int; maxScrollH: int; maxScrollV: int; mouseWheelEnabled: Boolean; multiline: Boolean; numLines: int; restrict: String; scrollH: int; scrollV: int; selectable: Boolean; selectionBeginIndex: int; selectionEndIndex: int; sharpness: Number; styleSheet: StyleSheet; text: String; textColor: uint; textHeight: Number; textWidth: Number; thickness: Number; type: String; wordWrap: Boolean; appendText() ; getCharBoundaries() : Rectangle; getCharIndexAtPoint(): int; getFirstCharInParagraph(): int; getImageReference(): DisplayObject; getLineIndexAtPoint(): int; getLineIndexOfChar(): int; getLineLength(): int; getLineMetric s(): TextLineMetrics; getLineOffset(): int; getLineText(): String; getParagraphLength(): int; getTextFormat(): TextFormat; replaceSelectedText() ; replaceText() ; setSelection() ; setTextFormat() ; TextField(): Object

DisplayObject **StaticText** — text: String

EventDispatcher **StyleSheet** — styleNames: Array; clear() ; getStyle(): Object; parseCSS() ; setStyle() ; StyleSheet(): Object; transform() : TextFormat

Object **TextFormat** — align: String; blockIndent: Object; bold: Object; bullet: Object; color: Object; font: String; indent: Object; italic: Object; kerning: Object; leading: Object; leftMargin: Object; letterSpacing: Object; rightMargin: Object; size: Object; tabStops: Array; target: String; underline: Object; url: String; TextFormat(): Object

Object **Font** — fontName: String; fontStyle: String; fontType: String; enum erateFonts() : Array; hasGlyphs() : Boolean; registerFont()

Object **FontType** — DEVICE: String; EMBEDDED: String

Object **FontStyle** — BOLD: String; BOLD_ITA LIC: String; ITALIC: String; REGULAR: String

Object **TextLineMetrics** — ascent: Number; descent: Number; height: Number; leading: Number; width: Number; x: Number; TextLineMetrics(): Object

Object **TextRenderer** — displayMode: String; maxLevel: int; setAdvancedAntiAliasingTable()

Object **TextSnapshot** — charCount: int; findText() : int; getSelected() : Boolean; getSelectedText() : String; getText() : String; getTextRunInfo() : Array; hitTestTextNearPos() : Number; setSelectC olor() ; setSelected()

Object **AntiAliasType** — ADVANCED: String; NORMAL: String

Object **CSMSettings** — fontSize: Number; insideCutoff: Number; outsideCutoff: Number; CSMSettings(): Object

Object **GridFitType** — NONE: String; PIXEL: String; SUBPIXEL: String

Object **TextColorType** — DARK_COLOR: String; LIGHT_COLOR: String

Object **TextDisplayMode** — CRT: String; DEFAULT: String; LCD: String

Object **TextFieldAutoSize** — CENTER: String; LEFT: String; NONE: String; RIGHT: String

Object **TextFieldType** — DYNAMIC: String; INPUT: String

Object **TextFormatAlign** — CENTER: String; JUSTIFY: String; LEFT: String; RIGHT: String

flash.net

Object **FileFilter** — description: String; extension: String; macType: String; FileFilter(): Object

EventDispatcher **FileReference** — creationDate: Date; creator: String; modificationDate: Date; name: String; size: uint; type: String; browse() : Boolean; cancel() ; download() ; FileReference(): Object; upload()

EventDispatcher **FileReferenceList** — fileList: Array; browse(): Boolean; FileReferenceList(): Object

«interface» **IDynamicPropertyOutput** — writeDynamicProperty()

«interface» **IDynamicPropertyWriter** — writeDynamicProperties()

Object **LocalConnection** — AMF0: uint; AMF3: uint; DEFAULT: uint; dynamicPropertyWriter: IDynamicPropertyWriter

EventDispatcher **NetConnection** — client: Object; connected: Boolean; connectedProxyType: String; defaultObjectEncoding: uint; objectEncoding: uint; proxyType: String; uri: String; usingTLS: Boolean; call() ; close() ; connect() ; NetConnection(): Object

EventDispatcher **NetStream** — bufferLength: Number; bufferTime: Number; bytesLoaded: uint; bytesTotal: uint; checkPolicyFile: Boolean; client: Object t; currentFPS: Number; liveDelay: Number; objectEncoding: uint; soundTransform : SoundTransform; time: Number; attachAudio() ; attachCamera() ; close() ; NetStream(): Object; pause() ; play() ; publish() ; resume() ; seek() ; send() ; togglePause()

EventDispatcher **LocalConnection** — client: Object; domain: String; allowDomain() ; allowInsecureDomain() ; close() ; connect() ; LocalConnection(): Object; send()

Object **Responder**

EventDispatcher **SharedObject** — client: Object; data: Object; defaultObjectEncoding: uint; fps: Number; objectEncoding: uint; size: uint; clear() ; close() ; connect() ; flush(): String; getLocal(): SharedObject; getRemote(): SharedObject; setDirty() ; setProperty()

Object **SharedObjectFlushStatus** — FLUSHED: String; PENDING: String

EventDispatcher **Socket** — bytesAvailable: uint; connected: Boolean; endian: String; objectEncoding: uint; close() ; connect() ; flush() ; readBoolean(): Boolean; readByte(): int; readBytes() ; readDouble(): Number; readFloat(): Number; readInt(): int; readMultiByte(): String; readObject(): Object; readShort(): int; readUnsignedByte(): uint; readUnsignedInt(): uint; readUnsignedShort(): uint; readUTF(): String; readUTFBytes(): String; Socket(): Object; writeBoolean() ; writeByte() ; writeBytes() ; writeDouble() ; writeFloat() ; writeInt() ; writeMultiBytes() ; writeO bject t() ; writeShort() ; writeUnsignedInt() ; writeUTF() ; writeUTFBytes()

EventDispatcher **URLLoader** — bytesLoaded: uint; bytesTotal: uint; data: ; dataFormat: String; close() ; load() ; URLLoader(): Object

Object **URLLoaderDataFormat** — BINARY: String; TEXT: String; VARIABLES: String

Object **URLRequest** — contentType: String; data: Object; method: String; requestHeaders: Array; url: String; URLRequest(): Object

Object **URLRequestHeader** — name: String; value: String; URLRequestHeader(): Object

Object **URLRequestMethod** — GET: String; POST: String

EventDispatcher **URLStream** — bytesAvailable: uint; connected: Boolean; endian: String; objectEncoding: uint; close() ; load() ; readBoolean(): Boolean; readByte(): int; readBytes() ; readDouble(): Number; readFloat(): Number; readInt(): int; readMultiByte(): String; readObject(): Object; readShort(): int; readUnsignedByte(): uint; readUnsignedInt(): uint; readUnsignedShort(): uint; readUTF(): String; readUTFBytes(): String

Object **URLVariables** — decode() ; toString() : String; URLVariables(): Object

EventDispatcher **URLVariables** — connected: Boolean; close() ; connect() ; send() ; XMLSocket(): Object

IMPLEMENTS flash.utils::IDataInput

IMPLEMENTS flash.utils::IDataOutput

flash.system

Object **ApplicationDomain** — currentDomain: ApplicationDomain; parentDomain: ApplicationDomain; ApplicationDomain(): Object; getDefinition(): Object

Object **Capabilities** — avHardwareDisable: Boolean; hasAccessibility: Boolean; hasAudio: Boolean; hasAudioEncoder: Boolean; hasEmbeddedVideo: Boolean; hasIME: Boolean; hasMP3: Boolean; hasPrinting: Boolean; hasScreenBroadcast: Boolean; hasScreenPlayback: Boolean; hasStreamingAudio: Boolean; hasStreamingVideo: Boolean; hasTLS: Boolean; hasVideoEncoder: Boolean; isDebugger: Boolean; language: String; localFileReadDisable: Boolean; manufacturer: String; os: String; pixelAspectRatio: Number; playerType: String; screenColor: String; screenDPI: Number; screenResolutionX: Number; screenResolutionY: Number; serverString: String; version: String

EventDispatcher **IME** — conversionMode: String; enabled: Boolean; doConversion() ; setCompositionString()

Object **IMEConversionMode** — ALPHANUMERIC_FULL: String; ALPHANUMERIC_HALF: String; CHINESE: String; JAPANESE_HIRAGANA : String; JAPANESE_KATAKANA _FULL: String; JAPANESE_KATAKANA _HALF: String; KOREAN: String; UNKNOWN: String

Object **LoaderContext** — applicationDomain: ApplicationDomain; checkPolicyFile: Boolean; securityDomain: SecurityDomain; LoaderContext(): Object

Object **SecurityDomain** — currentDomain: SecurityDomain

Object **Security** — exactSettings: Boolean; LOCAL_TRUSTED: String; LOCAL_WITH_FILE: String; LOCAL_WITH_NETWORK: String; REMOTE: String; sandboxType: String; allowDomain() ; allowInsecureDomain() ; loadPolicyFile() ; showSettings()

Object **System** — ime: IME; totalMemory: uint; useCodePage: Boolean; setClipboard()

Object **SecurityPanel** — CAMERA : String; DEFAULT: String; LOCAL_STORAGE: String; MICROPHONE: String; PRIVACY: String; SETTINGS_MANAGER: String

flash.printing

EventDispatcher **PrintJob** — orientation: String; pageHeight: int; pageWidth: int; paperHeight: int; paperWidth: int; addPage() ; PrintJob(): Object; send() ; start(): Boolean

Object **PrintJobOptions** — printAsBitmap: Boolean; PrintJobOptions(): Object

Object **PrintJobOrientation** — LANDSCAPE: String; PORTRAIT: String

flash.xml

Object **XMLNode** — attributes: Object; childNodes: Array; firstChild: XMLNode; lastChild: XMLNode; localName: String; namespaceURI: String; nextSibling: XMLNode; nodeName: String; nodeType: uint; nodeValue: String; parentNode: XMLNode; prefix: String; previousSibling: XMLNode; appendChild() ; cloneNode(): XMLNode; getNamespaceForPrefix(): String; getPrefixForNamespace() : String; hasChildNodes(): Boolean; insertBefore() ; removeNode() ; toString(): String; XMLNode(): Object

Object **XMLDocument** — docTypeDecl: Object; idMap: Object; ignoreWhite: Boolean; xmlDecl: Object; createElement(): XMLNode; createTextNode(): XMLNode; parseXML() ; toString() : String; XMLDocument(): Object

Object **XMLNodeType** — ELEMENT_NODE: uint; TEXT_NODE: uint

Top Level Classes

arguments — callee: Function; length: Number

Array — CASEINSENSITIVE: uint; DESCENDING: uint; length: uint; NUMERIC: uint; RETURNINDEXEDARRAY: uint; UNIQUESORT: uint; Array() : Object; Array() : Object; concat() : Array; every() : Boolean; filter() : Array; forEach() ; indexOf(): int; join() : String; lastIndexOf(): int; map() : Array; pop() : Object; push() : uint; reverse() : Array; shift() : Object; slice() : Array; some() : Boolean; sort() : Array; sortOn() : Array; splice() : Array; toLocaleString() : String; toString() : String; unshift() : uint

Boolean — Boolean() : Object; toString() : String; valueOf() : Boolean

Class

Date — date: Number; dateUTC: Number; day: Number; dayUTC: Number; fullYear: Number; fullYearUTC: Number; hours: Number; hoursUTC: Number; milliseconds: Number; milliscondsUTC: Number; minutes: Number; minutesUTC: Number; month: Number; monthUTC: Number; seconds: Number; secondsUTC: Number; time: Number; timezoneOffset: Number; Date() : Object; getDate(): Number; getDay(): Number; getFullYear(): Number; getHours() : Number; getMilliseconds() : Number; getMinutes() : Number; getMonth() : Number; getSeconds() : Number; getTime() : Number; getTimezoneOffset() : Number; getUTCDate() : Number; getUTCDay() : Number; getUTCFullYear() : Number; getUTCHours() : Number; getUTCMilliseconds() : Number; getUTCMinutes() : Number; getUTCMonth() : Number; getUTCSeconds() : Number; parse() : Number; setDate() : Number; setFullYear() : Number; setHours() : Number; setMilliseconds() : Number; setMinutes() : Number; setMonth() : Number; setSeconds() : Number; setTime() : Number; setUTCDate() : Number; setUTCFullYear() : Number; setUTCHours() : Number; setUTCMilliseconds() : Number; setUTCMinutes() : Number; setUTCMonth() : Number; setUTCSeconds() : Number; toDateString() : String; toLocaleDateString() : String; toLocaleString() : String; toLocaleTimeString() : String; toString() : String; toTimeString() : String; toUTCString() : String; UTC() : Number; valueOf() : Number

Function — apply() ; call()

int — MA X_VALUE: int; MIN_VALUE: int; int() : Object; toString() : String; valueOf() : int

Math — E: Number; LN10: Number; LN2: Number; LOG10E: Number; LOG2E: Number; PI: Number; SQRT1_2: Number; SQRT2: Number; abs() : Number; ac os() : Number; asin() : Number; atan() : Number; atan2() : Number; ceil() : Number; cos() : Number; exp() : Number; floor() : Number; log() : Number; max() : Number; min() : Number; pow() : Number; random() : Number; round() : Number; sin() : Number; sqrt() : Number; tan() : Number

Namespace — prefix: String; uri: String; Namespace() : Object; Namespace() : Object; toString() : String

Number — MA X_VA LUE: Number; MIN_VA LUE: Number; NaN: Number; NEGATIVE_INFINITY: Number; POSITIVE_INFINITY: Number; Number() : Object; toExponential() : String; toFixed() : String; toPrecision() : String; toString() : String; valueOf() : Number

Object — constructor: Object; prototype: Object; hasOwnProperty() : Boolean; isPrototypeOf() : Boolean; Object() : Object; propertyIsEnumerable() : Boolean; setPropertyIsEnumerable() ; toString() : String; valueOf() : Object

QName — localName: String; url: String; QName() : Object; QName() : Object; toString() : String

RegExp — dotall: Boolean; extended: Boolean; global: Boolean; ignoreCase: Boolean; lastIndex: Number; multiline: Boolean; source: String; exec() : Object; RegExp() : Object; test() : Boolean

String — length: int; charAt() : String; charCodeAt() : Number; concat() : String; fromCharCode() : String; indexOf() : int; lastIndexOf() : int; localeCompare() : int; match() : Array; replace() : String; search() : int; slice() : String; split() : Array; String() : Object; substr() : String; substring() : String; toLocaleLowerCase() : String; toLocaleUpperCase() : String; toLowerCase() : String; toUpperCase() : String; valueOf() : String

uint — MAX_VALUE: uint; MIN_VALUE: uint; toString() : String; uint() : Object; valueOf() : uint

XML — ignoreComments: Boolean; ignoreProcessingInstructions: Boolean; ignoreWhitespace: Boolean; prettyIndent: int; prettyPrinting: Boolean; addNamespace() : XML; appendChild() : XML; attribute() : XMLList; attributes() : XMLList; child() : XMLList; childIndex() : int; children() : XMLList; comments() : XMLList; contains() : Boolean; copy() : XML; defaultSettings() : Object; descendants() : XMLList; elements() : XMLList; hasComplexContent() : Boolean; hasOwnProperty() : Boolean; hasSimpleContent() : Boolean; inScopeNamespaces() : Array; insertChildAfter() : Object; insertChildBefore() : Object; length() : int; localName() : Object; name() : Object; namespace() : Object; namespaceDeclarations() : Array; nodeKind() : String; normalize() : XML; parent() : Object; prependChild() : XML; proc essingInstructions() : XMLList; propertyIsEnumerable() : Boolean; removeNamespace() : XML; replace() : XML; setChildren() : XML; setLocalName() ; setName() ; setNamespace() ; setSettings() ; settings() : Object; text() : XMLList; toString() : String; toXMLString() : String; valueOf() : XML; XML() : Object

XMLList — attribute() : XMLList; attributes() : XMLList; child() : XMLList; children() : XMLList; comments() : XMLList; contains() : Boolean; copy() : XMLList; descendants() : XMLList; elements() : XMLList; hasComplexContent() : Boolean; hasOwnProperty() : Boolean; hasSimpleContent() : Boolean; length() : int; normalize() : XMLList; parent