Continue

Javascript on pdf

pdf-lib Index Enumerations Classes Interfaces Type aliases Variables Functions Object literals Type aliases AppearanceMapping T AppearanceMapping: object Defined in api/form/appearances.ts:88 Appearance Provider Utility Types Type declaration: down? : T normal: T rollover? : T AppearanceOrMapping T AppearanceOrMapping: T | AppearanceMapping‹T› Defined in api/form/appearances.ts:90 AppearanceProviderFor T AppearanceProviderFor: T extends PDFCheckBox ? CheckBoxAppearanceProvider : T extends PDFRadioGroup ? RadioGroupAppearanceProvider : T extends PDFButton ? ButtonAppearanceProvider : T extends PDFDropdown ? DropdownAppearanceProvider : T extends PDFOptionList ? OptionListAppearanceProvider : T extends PDFTextField ? TextFieldAppearanceProvider : T extends PDFSignature ? SignatureAppearanceProvider : never Defined in api/form/appearances.ts:93 BoolViewerPrefKey T BoolViewerPrefKey: "HideToolbar" | "HideMenubar" | "HideWindowUI" | "FitWindow" | "CenterWindow" | "DisplayDocTitle" | "PickTrayByPDFSize" Defined in core/interactive/ViewerPreferences.ts:78 ButtonAppearanceProvider T ButtonAppearanceProvider: function Defined in api/form/appearances.ts:56 Type declaration: ▸ (button: PDFButton, widget: PDFWidgetAnnotation, font: PDFFont): AppearanceOrMapping‹PDFOperator[]› Parameters: NameType buttonPDFButton widgetPDFWidgetAnnotation fontPDFFont CheckBoxAppearanceProvider T CheckBoxAppearanceProvider: function Defined in api/form/appearances.ts:40 Appearance Provider Types Type declaration: ▸ (checkBox: PDFCheckBox, widget: PDFWidgetAnnotation): AppearanceOrMapping‹object› Parameters: NameType checkBoxPDFCheckBox widgetPDFWidgetAnnotation Color T Color: Grayscale | RGB | CMYK Defined in api/colors.ts:37 DropdownAppearanceProvider T DropdownAppearanceProvider: function Defined in api/form/appearances.ts:62 Type declaration: ▸ (dropdown: PDFDropdown, widget: PDFWidgetAnnotation, font: PDFFont): AppearanceOrMapping‹PDFOperator[]› Parameters: NameType dropdownPDFDropdown widgetPDFWidgetAnnotation fontPDFFont FontEmbedder T FontEmbedder: CustomFontEmbedder | StandardFontEmbedder Defined in api/PDFFont.ts:11 ImageEmbedder T ImageEmbedder: JpegEmbedder | PngEmbedder Defined in api/PDFImage.ts:6 NameViewerPrefKey T NameViewerPrefKey: "NonFullScreenPageMode" | "Direction" | "PrintScaling" | "Duplex" Defined in core/interactive/ViewerPreferences.ts:86 OptionListAppearanceProvider T OptionListAppearanceProvider: function Defined in api/form/appearances.ts:68 Type declaration: ▸ (optionList: PDFOptionList, widget: PDFWidgetAnnotation, font: PDFFont): AppearanceOrMapping‹PDFOperator[]› Parameters: NameType optionListPDFOptionList widgetPDFWidgetAnnotation fontPDFFont RadioGroupAppearanceProvider T RadioGroupAppearanceProvider: function Defined in api/form/appearances.ts:48 Type declaration: ▸ (radioGroup: PDFRadioGroup, widget: PDFWidgetAnnotation): AppearanceOrMapping‹object› Parameters: NameType radioGroupPDFRadioGroup widgetPDFWidgetAnnotation Rotation T Rotation: Radians | Degrees Defined in api/rotations.ts:18 SignatureAppearanceProvider T SignatureAppearanceProvider: function Defined in api/form/appearances.ts:80 Type declaration: ▸ (signature: PDFSignature, widget: PDFWidgetAnnotation, font: PDFFont): AppearanceOrMapping‹PDFOperator[]› Parameters: NameType signaturePDFSignature widgetPDFWidgetAnnotation fontPDFFont TextFieldAppearanceProvider T TextFieldAppearanceProvider: function Defined in api/form/appearances.ts:74 Type declaration: ▸ (textField: PDFTextField, widget: PDFWidgetAnnotation, font: PDFFont): AppearanceOrMapping‹PDFOperator[]› Parameters: NameType textFieldPDFTextField widgetPDFWidgetAnnotation fontPDFFont TransformationMatrix T TransformationMatrix: [number, number, number, number, number, number] Defined in types/matrix.ts:20 A transformation matrix according to section 8.3.3 Common Transformations of the PDF specification (page 117). To cite from the spec: Translations shall be specified as [1 0 0 1 tx ty], where tx and ty shall be the distances to translate the origin of the coordinate system in the horizontal and vertical dimensions, respectively. Scaling shall be obtained by [sx 0 0 sy 0 0]. This scales the coordinates so that 1 unit in the horizontal and vertical dimensions of the new coordinate system is the same size as sx and sy units, respectively, in the previous coordinate system. Rotations shall be produced by [cos(q) sin(q) -sin(q) cos(q) 0 0], which has the effect of rotating the coordinate system axes by an angle q counter clockwise. Skew shall be specified by [1 tan(a) tan(b) 1 0 0], which skews the x-axis by an angle a and the y axis by an angle b. Variables CMYK • CMYK: CMYK Defined in api/colors.ts:64 Degrees • Degrees: Degrees Defined in api/rotations.ts:30 Grayscale • Grayscale: Grayscale Defined in api/colors.ts:64 Const KAPPA • KAPPA: number = 4.0 * ((Math.sqrt(2) - 1.0) / 3.0) Defined in api/operations.ts:220 Const MAX_FONT_SIZE • MAX_FONT_SIZE: 500 = 500 Defined in api/text/layout.ts:31 Const MIN_FONT_SIZE • MIN_FONT_SIZE: 4 = 4 Defined in api/text/layout.ts:30 RGB • RGB: RGB Defined in api/colors.ts:64 Radians • Radians: Radians Defined in api/rotations.ts:30 Const colorRegex • colorRegex: RegExp‹› = /(\d.\d+|\d+)[\0\t\f\r\ ](\d.\d+|\d+)?[\0\t\f\r\ ](\d.\d+|\d+)?[\0\t\f\r\ ](\d*.\d+|\d+)?[\0\t\f\r\ ]+(g|rg|k)/ Defined in api/form/appearances.ts:130 cos • cos: cos Defined in api/operators.ts:18 Const identityMatrix • identityMatrix: TransformationMatrix = [1, 0, 0, 1, 0, 0] Defined in types/matrix.ts:29 sin • sin: sin Defined in api/operators.ts:18 tan • tan: tan Defined in api/operators.ts:18 Const tfRegex • tfRegex: RegExp‹› = //([^\0\t\f\r\ ]+)[\0\t\f\r\ ]+(\d*.\d+|\d+)[\0\t\f\r\ ]+Tf/ Defined in api/form/appearances.ts:115 Functions Const addFieldToParent ▸ addFieldToParent(__namedParameters: [PDFAcroForm‹› | PDFAcroNonTerminal‹›, undefined | PDFRef‹›], __namedParameters: [PDFAcroField, PDFRef], partialName: string): void Defined in api/form/PDFForm.ts:838 Parameters: NameType __namedParameters[PDFAcroForm‹› | PDFAcroNonTerminal‹›, undefined | PDFRef‹›] __namedParameters[PDFAcroField, PDFRef] partialNamestring Returns: void Const adjustDimsForRotation ▸ adjustDimsForRotation(dims: object, degreeAngle: number): object Defined in api/rotations.ts:56 Parameters: ■ dims: object NameType heightnumber widthnumber ■Default value degreeAngle: number= 0 Returns: object height: number = dims.width width: number = dims.height Const appendBezierCurve ▸ appendBezierCurve(x1: number | PDFNumber, y1: number | PDFNumber, x2: number | PDFNumber, y2: number | PDFNumber, x3: number | PDFNumber, y3: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:119 Parameters: NameType x1number | PDFNumber y1number | PDFNumber x2number | PDFNumber y2number | PDFNumber x3number | PDFNumber y3number | PDFNumber Returns: PDFOperator‹› Const appendQuadraticCurve ▸ appendQuadraticCurve(x1: number | PDFNumber, y1: number | PDFNumber, x2: number | PDFNumber, y2: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:136 Parameters: NameType x1number | PDFNumber y1number | PDFNumber x2number | PDFNumber y2number | PDFNumber Returns: PDFOperator‹› Const asEnum ▸ asEnum(rawValue: T | undefined, enumType: U): U[keyof U] | undefined Defined in core/interactive/ViewerPreferences.ts:14 Type parameters: ■ T: string | number ■ U: object Parameters: NameType rawValueT | undefined enumTypeU Returns: U[keyof U] | undefined Const asNumber ▸ asNumber(num: number | PDFNumber): number Defined in api/objects.ts:9 Parameters: NameType numnumber | PDFNumber Returns: number Const asPDFName ▸ asPDFName(name: string | PDFName): PDFName‹› Defined in api/objects.ts:3 Parameters: NameType namestring | PDFName Returns: PDFName‹› Const asPDFNumber ▸ asPDFNumber(num: number | PDFNumber): PDFNumber‹› Defined in api/objects.ts:6 Parameters: NameType numnumber | PDFNumber Returns: PDFNumber‹› Const assertFieldAppearanceOptions ▸ assertFieldAppearanceOptions(options?: FieldAppearanceOptions): void Defined in api/form/PDFField.ts:44 Parameters: NameType options?FieldAppearanceOptions Returns: void assertIsLiteralOrHexString ▸ assertIsLiteralOrHexString(pdfObject: PDFObject): asserts pdfObject is PDFHexString | PDFString Defined in api/PDFDocument.ts:1386 Parameters: NameType pdfObjectPDFObject Returns: asserts pdfObject is PDFHexString | PDFString Const beginMarkedContent ▸ beginMarkedContent(tag: string | PDFName): PDFOperator‹› Defined in api/operators.ts:357 Parameters: NameType tagstring | PDFName Returns: PDFOperator‹› Const beginText ▸ beginText(): PDFOperator‹› Defined in api/operators.ts:208 Returns: PDFOperator‹› Const clip ▸ clip(): PDFOperator‹› Defined in api/operators.ts:13 Returns: PDFOperator‹› Const clipEvenOdd ▸ clipEvenOdd(): PDFOperator‹› Defined in api/operators.ts:14 Returns: PDFOperator‹› Const closePath ▸ closePath(): PDFOperator‹› Defined in api/operators.ts:149 Returns: PDFOperator‹› Const cmyk ▸ cmyk(cyan: number, magenta: number, yellow: number, key: number): CMYK Defined in api/colors.ts:51 Parameters: NameType cyannumber magentanumber yellownumber keynumber Returns: CMYK Const colorToComponents ▸ colorToComponents(color: Color): number[] Defined in api/colors.ts:100 Parameters: Returns: number[] Const componentsToColor ▸ componentsToColor(comps?: number[], scale: number): undefined | Grayscale | RGB | CMYK Defined in api/colors.ts:81 Parameters: NameTypeDefault comps?number[]- scalenumber1 Returns: undefined | Grayscale | RGB | CMYK Const computeCombedFontSize ▸ computeCombedFontSize(line: string, font: PDFFont, bounds: LayoutBounds, cellCount: number): number Defined in api/text/layout.ts:58 Parameters: NameType linestring fontPDFFont boundsLayoutBounds cellCountnumber Returns: number Const computeFontSize ▸ computeFontSize(lines: string[], font: PDFFont, bounds: LayoutBounds): number Defined in api/text/layout.ts:33 Parameters: NameType linesstring[] fontPDFFont boundsLayoutBounds Returns: number Const concatTransformationMatrix ▸ concatTransformationMatrix(a: number | PDFNumber, b: number | PDFNumber, c: number | PDFNumber, d: number | PDFNumber, e: number | PDFNumber, f: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:20 Parameters: NameType anumber | PDFNumber bnumber | PDFNumber cnumber | PDFNumber dnumber | PDFNumber enumber | PDFNumber fnumber | PDFNumber Returns: PDFOperator‹› Const convertToPDFField ▸ convertToPDFField(field: PDFAcroField, ref: PDFRef, doc: PDFDocument): PDFField | undefined Defined in api/form/PDFForm.ts:796 Parameters: NameType fieldPDFAcroField refPDFRef docPDFDocument Returns: PDFField | undefined Const defaultButtonAppearanceProvider ▸ defaultButtonAppearanceProvider(button: PDFButton‹›, widget: PDFWidgetAnnotation‹›, font: PDFFont‹›): object Defined in api/form/appearances.ts:329 Parameters: NameType buttonPDFButton‹› widgetPDFWidgetAnnotation‹› fontPDFFont‹› Returns: object down: PDFOperator‹›[] = [...rotate, ...drawButton({ ...options, color: downBackgroundColor, textLines: [downLayout.line], }),] normal: PDFOperator‹›[] = [...rotate, ...drawButton({ ...options, color: normalBackgroundColor, textLines: [normalLayout.line], }),] Const defaultCheckBoxAppearanceProvider ▸ defaultCheckBoxAppearanceProvider(checkBox: PDFCheckBox‹›, widget: PDFWidgetAnnotation‹›): object Defined in api/form/appearances.ts:166 Parameters: NameType checkBoxPDFCheckBox‹› widgetPDFWidgetAnnotation‹› Returns: object off: PDFOperator‹›[] = [...rotate, ...drawCheckBox({ ...options, color: downBackgroundColor, filled: false, }),] on: PDFOperator‹›[] = [...rotate, ...drawCheckBox({ ...options, color: downBackgroundColor, filled: true, }),] off: PDFOperator‹›[] = [...rotate, ...drawCheckBox({ ...options, color: normalBackgroundColor, filled: false, }),] on: PDFOperator‹›[] = [...rotate, ...drawCheckBox({ ...options, color: normalBackgroundColor, filled: true, }),] Const defaultDropdownAppearanceProvider ▸ defaultDropdownAppearanceProvider(dropdown: PDFDropdown‹›, widget: PDFWidgetAnnotation‹›, font: PDFFont‹›): PDFOperator‹›[] Defined in api/form/appearances.ts:511 Parameters: NameType dropdownPDFDropdown‹› widgetPDFWidgetAnnotation‹› fontPDFFont‹› Returns: PDFOperator‹›[] Const defaultOptionListAppearanceProvider ▸ defaultOptionListAppearanceProvider(optionList: PDFOptionList‹›, widget: PDFWidgetAnnotation‹›, font: PDFFont‹›): PDFOperator‹›[] Defined in api/form/appearances.ts:578 Parameters: Returns: PDFOperator‹›[] Const defaultRadioGroupAppearanceProvider ▸ defaultRadioGroupAppearanceProvider(radioGroup: PDFRadioGroup‹›, widget: PDFWidgetAnnotation‹›): object Defined in api/form/appearances.ts:248 Parameters: NameType radioGroupPDFRadioGroup‹› widgetPDFWidgetAnnotation‹› Returns: object off: PDFOperator‹›[] = [...rotate, ...drawRadioButton({ ...options, color: downBackgroundColor, filled: false, }),] on: PDFOperator‹›[] = [...rotate, ...drawRadioButton({ ...options, color: downBackgroundColor, filled: true, }),] off: PDFOperator‹›[] = [...rotate, ...drawRadioButton({ ...options, color: normalBackgroundColor, filled: false, }),] on: PDFOperator‹›[] = [...rotate, ...drawRadioButton({ ...options, color: normalBackgroundColor, filled: true, }),] Const defaultTextFieldAppearanceProvider ▸ defaultTextFieldAppearanceProvider(textField: PDFTextField‹›, widget: PDFWidgetAnnotation‹›, font: PDFFont‹›): PDFOperator‹›[] Defined in api/form/appearances.ts:419 Parameters: NameType textFieldPDFTextField‹› widgetPDFWidgetAnnotation‹› fontPDFFont‹› Returns: PDFOperator‹›[] Const degrees ▸ degrees(degreeAngle: number): Degrees Defined in api/rotations.ts:25 Parameters: NameType degreeAnglenumber Returns: Degrees Const degreesToRadians ▸ degreesToRadians(degree: number): number Defined in api/rotations.ts:32 Parameters: Returns: number Const drawButton ▸ drawButton(options: object): PDFOperator‹›[] Defined in api/operations.ts:547 Parameters: ■ options: object NameType borderColorColor | undefined borderWidthnumber | PDFNumber colorColor | undefined fontstring | PDFName fontSizenumber | PDFNumber heightnumber | PDFNumber textColorColor textLinesobject[] widthnumber | PDFNumber xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹›[] Const drawCheckBox ▸ drawCheckBox(options: object): PDFOperator‹›[] Defined in api/operations.ts:463 Parameters: ■ options: object NameType borderColorColor | undefined borderWidthnumber | PDFNumber colorColor | undefined filledboolean heightnumber | PDFNumber markColorColor | undefined thicknessnumber | PDFNumber widthnumber | PDFNumber xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹›[] Const drawCheckMark ▸ drawCheckMark(options: object): PDFOperator‹›[] Defined in api/operations.ts:381 Parameters: ■ options: object NameType colorColor | undefined sizenumber | PDFNumber thicknessnumber | PDFNumber xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹›[] Const drawEllipse ▸ drawEllipse(options: object): PDFOperator‹›[] Defined in api/operations.ts:288 Parameters: ■ options: object NameType borderColorColor | undefined borderDashArray?number | PDFNumber‹›[] borderDashPhase?number | PDFNumber borderLineCap?LineCapStyle borderWidthnumber | PDFNumber colorColor | undefined graphicsState?string | PDFName rotate?Rotation xnumber | PDFNumber xScalenumber | PDFNumber ynumber | PDFNumber yScalenumber | PDFNumber Returns: PDFOperator‹›[] Const drawEllipseCurves ▸ drawEllipseCurves(config: object): PDFOperator[] Defined in api/operations.ts:255 Parameters: ■ config: object NameType rotateRotation xnumber | PDFNumber xScalenumber | PDFNumber ynumber | PDFNumber yScalenumber | PDFNumber Returns: PDFOperator[] Const drawEllipsePath ▸ drawEllipsePath(config: object): PDFOperator[] Defined in api/operations.ts:223 deprecated Parameters: ■ config: object NameType xnumber | PDFNumber xScalenumber | PDFNumber ynumber | PDFNumber yScalenumber | PDFNumber Returns: PDFOperator[] Const drawImage ▸ drawImage(name: string | PDFName, options: object): PDFOperator[] Defined in api/operations.ts:106 Parameters: ■ name: string | PDFName ■ options: object NameType graphicsState?string | PDFName heightnumber | PDFNumber rotateRotation widthnumber | PDFNumber xnumber | PDFNumber xSkewRotation ynumber | PDFNumber ySkewRotation Returns: PDFOperator[] Const drawLine ▸ drawLine(options: object): PDFOperator‹›[] Defined in api/operations.ts:154 Parameters: ■ options: object NameType colorColor | undefined dashArray?number | PDFNumber‹›[] dashPhase?number | PDFNumber endobject graphicsState?string | PDFName lineCap?LineCapStyle startobject thicknessnumber | PDFNumber Returns: PDFOperator‹›[] Const drawLinesOfText ▸ drawLinesOfText(lines: PDFHexString[], options: DrawLinesOfTextOptions): PDFOperator[] Defined in api/operations.ts:78 Parameters: NameType linesPDFHexString[] optionsDrawLinesOfTextOptions Returns: PDFOperator[] Const drawObject ▸ drawObject(name: string | PDFName): PDFOperator‹› Defined in api/operators.ts:296 Parameters: NameType namestring | PDFName Returns: PDFOperator‹› Const drawOptionList ▸ drawOptionList(options: object): PDFOperator‹›[] Defined in api/operations.ts:703 Parameters: ■ options: object NameType borderColorColor | undefined borderWidthnumber | PDFNumber colorColor | undefined fontstring | PDFName fontSizenumber | PDFNumber heightnumber | PDFNumber lineHeightnumber | PDFNumber paddingnumber | PDFNumber selectedColorColor selectedLinesnumber[] textColorColor textLinesobject[] widthnumber | PDFNumber xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹›[] Const drawPage ▸ drawPage(name: string | PDFName, options: object): PDFOperator[] Defined in api/operations.ts:130 Parameters: ■ name: string | PDFName ■ options: object NameType graphicsState?string | PDFName rotateRotation xnumber | PDFNumber xScalenumber | PDFNumber xSkewRotation ynumber | PDFNumber yScalenumber | PDFNumber ySkewRotation Returns: PDFOperator[] Const drawRadioButton ▸ drawRadioButton(options: object): PDFOperator‹›[] Defined in api/operations.ts:506 Parameters: ■ options: object NameType borderColorColor | undefined borderWidthnumber | PDFNumber colorColor | undefined dotColorColor | undefined filledboolean heightnumber | PDFNumber widthnumber | PDFNumber xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹›[] Const drawRectangle ▸ drawRectangle(options: object): PDFOperator‹›[] Defined in api/operations.ts:178 Parameters: ■ options: object NameType borderColorColor | undefined borderDashArray?number | PDFNumber‹›[] borderDashPhase?number | PDFNumber borderLineCap?LineCapStyle borderWidthnumber | PDFNumber colorColor | undefined graphicsState?string | PDFName heightnumber | PDFNumber rotateRotation widthnumber | PDFNumber xnumber | PDFNumber xSkewRotation ynumber | PDFNumber ySkewRotation Returns: PDFOperator‹›[] Const drawSvgPath ▸ drawSvgPath(path: string, options: object): PDFOperator‹›[] Defined in api/operations.ts:337 Parameters: ■ path: string ■ options: object NameType borderColorColor | undefined borderDashArray?number | PDFNumber‹›[] borderDashPhase?number | PDFNumber borderLineCap?LineCapStyle borderWidthnumber | PDFNumber colorColor | undefined graphicsState?string | PDFName rotate?Rotation scalenumber | PDFNumber | undefined xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹›[] Const drawText ▸ drawText(line: PDFHexString, options: DrawTextOptions): PDFOperator[] Defined in api/operations.ts:52 Parameters: NameType linePDFHexString optionsDrawTextOptions Returns: PDFOperator[] Const drawTextField ▸ drawTextField(options: object): PDFOperator‹›[] Defined in api/operations.ts:628 Parameters: ■ options: object NameType borderColorColor | undefined borderWidthnumber | PDFNumber colorColor | undefined fontstring | PDFName fontSizenumber | PDFNumber heightnumber | PDFNumber paddingnumber | PDFNumber textColorColor textLinesobject[] widthnumber | PDFNumber xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹›[] Const drawTextLines ▸ drawTextLines(lines: object[], options: DrawTextLinesOptions): PDFOperator[] Defined in api/operations.ts:599 Parameters: NameType linesobject[] optionsDrawTextLinesOptions Returns: PDFOperator[] Const endMarkedContent ▸ endMarkedContent(): PDFOperator‹› Defined in api/operators.ts:360 Returns: PDFOperator‹› Const endPath ▸ endPath(): PDFOperator‹› Defined in api/operators.ts:192 Returns: PDFOperator‹› Const endText ▸ endText(): PDFOperator‹› Defined in api/operators.ts:209 Returns: PDFOperator‹› Const fill ▸ fill(): PDFOperator‹› Defined in api/operators.ts:188 Returns: PDFOperator‹› Const fillAndStroke ▸ fillAndStroke(): PDFOperator‹› Defined in api/operators.ts:190 Returns: PDFOperator‹› Const getDefaultColor ▸ getDefaultColor(field: object): undefined | Grayscale | RGB | CMYK Defined in api/form/appearances.ts:132 Parameters: ■ field: object NameType getDefaultAppearance Returns: undefined | Grayscale | RGB | CMYK Const getDefaultFontSize ▸ getDefaultFontSize(field: object): undefined | number Defined in api/form/appearances.ts:117 Parameters: ■ field: object NameType getDefaultAppearance Returns: undefined | number Const grayscale ▸ grayscale(gray: number): Grayscale Defined in api/colors.ts:39 Parameters: Returns: Grayscale Const lastIndexOfWhitespace ▸ lastIndexOfWhitespace(line: string): undefined | number Defined in api/text/layout.ts:100 Parameters: Returns: undefined | number Const layoutCombedText ▸ layoutCombedText(text: string, __namedParameters: object): CombedTextLayout Defined in api/text/layout.ts:220 Parameters: ■ text: string ■ __namedParameters: object Returns: CombedTextLayout Const layoutMultilineText ▸ layoutMultilineText(text: string, __namedParameters: object): MultilineTextLayout Defined in api/text/layout.ts:135 Parameters: ■ text: string ■ __namedParameters: object Returns: MultilineTextLayout Const layoutSinglelineText ▸ layoutSinglelineText(text: string, __namedParameters: object): SinglelineTextLayout Defined in api/text/layout.ts:293 Parameters: ■ text: string ■ __namedParameters: object Returns: SinglelineTextLayout Const lineTo ▸ lineTo(xPos: number | PDFNumber, yPos: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:154 Parameters: NameType xPosnumber | PDFNumber yPosnumber | PDFNumber Returns: PDFOperator‹› Const moveText ▸ moveText(x: number | PDFNumber, y: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:198 Parameters: NameType xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹› Const moveTo ▸ moveTo(xPos: number | PDFNumber, yPos: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:151 Parameters: NameType xPosnumber | PDFNumber yPosnumber | PDFNumber Returns: PDFOperator‹› Const nextLine ▸ nextLine(): PDFOperator‹› Defined in api/operators.ts:196 Returns: PDFOperator‹› Const normalizeAppearance ▸ normalizeAppearance(appearance: T | AppearanceMapping‹T›): AppearanceMapping‹T› Defined in api/form/appearances.ts:105 Appearance Provider Functions Type parameters: ■ T Parameters: NameType appearanceT | AppearanceMapping‹T› Returns: AppearanceMapping‹T› Const popGraphicsState ▸ popGraphicsState(): PDFOperator‹› Defined in api/operators.ts:112 Returns: PDFOperator‹› Const pushGraphicsState ▸ pushGraphicsState(): PDFOperator‹› Defined in api/operators.ts:110 Returns: PDFOperator‹› Const radians ▸ radians(radianAngle: number): Radians Defined in api/rotations.ts:20 Parameters: NameType radianAnglenumber Returns: Radians Const radiansToDegrees ▸ radiansToDegrees(radian: number): number Defined in api/rotations.ts:33 Parameters: Returns: number Const rectangle ▸ rectangle(xPos: number | PDFNumber, yPos: number | PDFNumber, width: number | PDFNumber, height: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:163 Parameters: NameTypeDescription xPosnumber | PDFNumberx coordinate for the lower left corner of the rectangle yPosnumber | PDFNumbery coordinate for the lower left corner of the rectangle widthnumber | PDFNumberwidth of the rectangle heightnumber | PDFNumberheight of the rectangle Returns: PDFOperator‹› Const reduceRotation ▸ reduceRotation(degreeAngle: number): 180 | 0 | 90 | 270 Defined in api/rotations.ts:47 Parameters: NameTypeDefault degreeAnglenumber0 Returns: 180 | 0 | 90 | 270 Const restoreDashPattern ▸ restoreDashPattern(): PDFOperator‹› Defined in api/operators.ts:87 Returns: PDFOperator‹› Const rgb ▸ rgb(red: number, green: number, blue: number): RGB Defined in api/colors.ts:44 Parameters: NameType rednumber greennumber bluenumber Returns: RGB Const rotateAndSkewTextDegreesAndTranslate ▸ rotateAndSkewTextDegreesAndTranslate(rotationAngle: number | PDFNumber, xSkewAngle: number | PDFNumber, ySkewAngle: number | PDFNumber, x: number | PDFNumber, y: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:279 Parameters: NameType rotationAnglenumber | PDFNumber xSkewAnglenumber | PDFNumber ySkewAnglenumber | PDFNumber xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹› Const rotateAndSkewTextRadiansAndTranslate ▸ rotateAndSkewTextRadiansAndTranslate(rotationAngle: number | PDFNumber, xSkewAngle: number | PDFNumber, ySkewAngle: number | PDFNumber, x: number | PDFNumber, y: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:263 Parameters: NameType rotationAnglenumber | PDFNumber xSkewAnglenumber | PDFNumber ySkewAnglenumber | PDFNumber xnumber | PDFNumber ynumber | PDFNumber Returns: PDFOperator‹› Const rotateDegrees ▸ rotateDegrees(angle: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:53 Parameters: NameType anglenumber | PDFNumber Returns: PDFOperator‹› Const rotateInPlace ▸ rotateInPlace(options: object): PDFOperator‹›[] Defined in api/operations.ts:440 Parameters: ■ options: object NameType heightnumber | PDFNumber rotation0 | 90 | 180 | 270 widthnumber | PDFNumber Returns: PDFOperator‹›[] Const rotateRadians ▸ rotateRadians(angle: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:43 Parameters: NameType anglenumber | PDFNumber Returns: PDFOperator‹› Const rotateRectangle ▸ rotateRectangle(rectangle: object, borderWidth: number, degreeAngle: number): object Defined in api/rotations.ts:66 Parameters: ■ rectangle: object NameType heightnumber widthnumber xnumber ynumber ■Default value borderWidth: number= 0 ■Default value degreeAngle: number= 0 Returns: object height: number = h width: number = w x: number = x - b y: number = y - b Const scale ▸ scale(xPos: number | PDFNumber, yPos: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:40 Parameters: NameType xPosnumber | PDFNumber yPosnumber | PDFNumber Returns: PDFOperator‹› Const setCharacterSpacing ▸ setCharacterSpacing(spacing: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:216 Parameters: NameType spacingnumber | PDFNumber Returns: PDFOperator‹› Const setCharacterSqueeze ▸ setCharacterSqueeze(squeeze: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:223 Parameters: NameTypeDescription squeezenumber | PDFNumberhorizontal character spacing Returns: PDFOperator‹› Const setDashPattern ▸ setDashPattern(dashArray: number | PDFNumber‹›[], dashPhase: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:78 Parameters: NameType dashArraynumber | PDFNumber‹›[] dashPhasenumber | PDFNumber Returns: PDFOperator‹› Const setFillingCmykColor ▸ setFillingCmykColor(cyan: number | PDFNumber, magenta: number | PDFNumber, yellow: number | PDFNumber, key: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:329 Parameters: NameType cyannumber | PDFNumber magentanumber | PDFNumber yellownumber | PDFNumber keynumber | PDFNumber Returns: PDFOperator‹› Const setFillingColor ▸ setFillingColor(color: Color): PDFOperator‹› Defined in api/colors.ts:67 Parameters: Returns: PDFOperator‹› Const setFillingGrayscaleColor ▸ setFillingGrayscaleColor(gray: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:301 Parameters: NameType graynumber | PDFNumber Returns: PDFOperator‹› Const setFillingRgbColor ▸ setFillingRgbColor(red: number | PDFNumber, green: number | PDFNumber, blue: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:307 Parameters: NameType rednumber | PDFNumber greennumber | PDFNumber bluenumber | PDFNumber Returns: PDFOperator‹› Const setFontAndSize ▸ setFontAndSize(name: string | PDFName, size: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:211 Parameters: NameType namestring | PDFName sizenumber | PDFNumber Returns: PDFOperator‹› Const setGraphicsState ▸ setGraphicsState(state: string | PDFName): PDFOperator‹› Defined in api/operators.ts:107 Parameters: NameType statestring | PDFName Returns: PDFOperator‹› Const setLineCap ▸ setLineCap(style: LineCapStyle): PDFOperator‹› Defined in api/operators.ts:95 Parameters: NameType styleLineCapStyle Returns: PDFOperator‹› Const setLineHeight ▸ setLineHeight(lineHeight: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:226 Parameters: NameType lineHeightnumber | PDFNumber Returns: PDFOperator‹› Const setLineJoin ▸ setLineJoin(style: LineJoinStyle): PDFOperator‹› Defined in api/operators.ts:104 Parameters: NameType styleLineJoinStyle Returns: PDFOperator‹› Const setLineWidth ▸ setLineWidth(width: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:114 Parameters: NameType widthnumber | PDFNumber Returns: PDFOperator‹› Const setStrokingCmykColor ▸ setStrokingCmykColor(cyan: number | PDFNumber, magenta: number | PDFNumber, yellow: number | PDFNumber, key: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:342 Parameters: NameType cyannumber | PDFNumber magentanumber | PDFNumber yellownumber | PDFNumber keynumber | PDFNumber Returns: PDFOperator‹› Const setStrokingColor ▸ setStrokingColor(color: Color): PDFOperator‹› Defined in api/colors.ts:74 Parameters: Returns: PDFOperator‹› Const setStrokingGrayscaleColor ▸ setStrokingGrayscaleColor(gray: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:304 Parameters: NameType graynumber | PDFNumber Returns: PDFOperator‹› Const setStrokingRgbColor ▸ setStrokingRgbColor(red: number | PDFNumber, green: number | PDFNumber, blue: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:318 Parameters: NameType rednumber | PDFNumber greennumber | PDFNumber bluenumber | PDFNumber Returns: PDFOperator‹› Const setTextMatrix ▸ setTextMatrix(a: number | PDFNumber, b: number | PDFNumber, c: number | PDFNumber, d: number | PDFNumber, e: number | PDFNumber, f: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:246 Parameters: NameType anumber | PDFNumber bnumber | PDFNumber cnumber | PDFNumber dnumber | PDFNumber enumber | PDFNumber fnumber | PDFNumber Returns: PDFOperator‹› Const setTextRenderingMode ▸ setTextRenderingMode(mode: TextRenderingMode): PDFOperator‹› Defined in api/operators.ts:243 Parameters: NameType modeTextRenderingMode Returns: PDFOperator‹› Const setTextRise ▸ setTextRise(rise: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:229 Parameters: NameType risenumber | PDFNumber Returns: PDFOperator‹› Const setWordSpacing ▸ setWordSpacing(spacing: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:219 Parameters: NameType spacingnumber | PDFNumber Returns: PDFOperator‹› Const showText ▸ showText(text: PDFHexString): PDFOperator‹› Defined in api/operators.ts:203 Parameters: NameType textPDFHexString Returns: PDFOperator‹› Const skewDegrees ▸ skewDegrees(xSkewAngle: number | PDFNumber, ySkewAngle: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:69 Parameters: NameType xSkewAnglenumber | PDFNumber ySkewAnglenumber | PDFNumber Returns: PDFOperator‹› Const skewRadians ▸ skewRadians(xSkewAngle: number | PDFNumber, ySkewAngle: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:56 Parameters: NameType xSkewAnglenumber | PDFNumber ySkewAnglenumber | PDFNumber Returns: PDFOperator‹› Const splitFieldName ▸ splitFieldName(fullyQualifiedName: string): object Defined in api/form/PDFForm.ts:815 Parameters: NameType fullyQualifiedNamestring Returns: object nonTerminal: string[] = parts.slice(0, parts.length - 1) terminal: string = parts[parts.length - 1] Const splitOutLines ▸ splitOutLines(input: string, maxWidth: number, font: PDFFont, fontSize: number): object Defined in api/text/layout.ts:107 Parameters: NameType inputstring maxWidthnumber fontPDFFont fontSizenumber Returns: object encoded: PDFHexString‹› line: string remainder: undefined | string width: number Const square ▸ square(xPos: number, yPos: number, size: number): PDFOperator‹› Defined in api/operators.ts:181 Parameters: NameTypeDescription xPosnumberx coordinate for the lower left corner of the square yPosnumbery coordinate for the lower left corner of the square sizenumberwidth and height of the square Returns: PDFOperator‹› Const stroke ▸ stroke(): PDFOperator‹› Defined in api/operators.ts:186 Returns: PDFOperator‹› Const toDegrees ▸ toDegrees(rotation: Rotation): number Defined in api/rotations.ts:42 Parameters: NameType rotationRotation Returns: number Const toRadians ▸ toRadians(rotation: Rotation): number Defined in api/rotations.ts:36 Parameters: NameType rotationRotation Returns: number Const translate ▸ translate(xPos: number | PDFNumber, yPos: number | PDFNumber): PDFOperator‹› Defined in api/operators.ts:37 Parameters: NameType xPosnumber | PDFNumber yPosnumber | PDFNumber Returns: PDFOperator‹› Const updateDefaultAppearance ▸ updateDefaultAppearance(field: object, color: Color, font?: PDFFont, fontSize: number): void Defined in api/form/appearances.ts:153 Parameters: ■ field: object NameType setDefaultAppearance ■ color: Color ■Optional font: PDFFont ■Default value fontSize: number= 0 Returns: void Object literals Const PageSizes ■ PageSizes: object Defined in api/sizes.ts:1 2A0 • 2A0: [number, number] = [3370.39, 4767.87] as [number, number] Defined in api/sizes.ts:3 4A0 • 4A0: [number, number] = [4767.87, 6740.79] as [number, number] Defined in api/sizes.ts:2 A0 • A0: [number, number] = [2383.94, 3370.39] as [number, number] Defined in api/sizes.ts:4 A1 • A1: [number, number] = [1683.78, 2383.94] as [number, number] Defined in api/sizes.ts:5 A10 • A10: [number, number] = [73.7, 104.88] as [number, number] Defined in api/sizes.ts:14 A2 • A2: [number, number] = [1190.55, 1683.78] as [number, number] Defined in api/sizes.ts:6 A3 • A3: [number, number] = [841.89, 1190.55] as [number, number] Defined in api/sizes.ts:7 A4 • A4: [number, number] = [595.28, 841.89] as [number, number] Defined in api/sizes.ts:8 A5 • A5: [number, number] = [419.53, 595.28] as [number, number] Defined in api/sizes.ts:9 A6 • A6: [number, number] = [297.64, 419.53] as [number, number] Defined in api/sizes.ts:10 A7 • A7: [number, number] = [209.76, 297.64] as [number, number] Defined in api/sizes.ts:11 A8 • A8: [number, number] = [147.4, 209.76] as [number, number] Defined in api/sizes.ts:12 A9 • A9: [number, number] = [104.88, 147.4] as [number, number] Defined in api/sizes.ts:13 B0 • B0: [number, number] = [2834.65, 4008.19] as [number, number] Defined in api/sizes.ts:15 B1 • B1: [number, number] = [2004.09, 2834.65] as [number, number] Defined in api/sizes.ts:16 B10 • B10: [number, number] = [87.87, 124.72] as [number, number] Defined in api/sizes.ts:25 B2 • B2: [number, number] = [1417.32, 2004.09] as [number, number] Defined in api/sizes.ts:17 B3 • B3: [number, number] = [1000.63, 1417.32] as [number, number] Defined in api/sizes.ts:18 B4 • B4: [number, number] = [708.66, 1000.63] as [number, number] Defined in api/sizes.ts:19 B5 • B5: [number, number] = [498.9, 708.66] as [number, number] Defined in api/sizes.ts:20 B6 • B6: [number, number] = [354.33, 498.9] as [number, number] Defined in api/sizes.ts:21 B7 • B7: [number, number] = [249.45, 354.33] as [number, number] Defined in api/sizes.ts:22 B8 • B8: [number, number] = [175.75, 249.45] as [number, number] Defined in api/sizes.ts:23 B9 • B9: [number, number] = [124.72, 175.75] as [number, number] Defined in api/sizes.ts:24 C0 • C0: [number, number] = [2599.37, 3676.54] as [number, number] Defined in api/sizes.ts:26 C1 • C1: [number, number] = [1836.85, 2599.37] as [number, number] Defined in api/sizes.ts:27 C10 • C10: [number, number] = [79.37, 113.39] as [number, number] Defined in api/sizes.ts:36 C2 • C2: [number, number] = [1298.27, 1836.85] as [number, number] Defined in api/sizes.ts:28 C3 • C3: [number, number] = [918.43, 1298.27] as [number, number] Defined in api/sizes.ts:29 C4 • C4: [number, number] = [649.13, 918.43] as [number, number] Defined in api/sizes.ts:30 C5 • C5: [number, number] = [459.21, 649.13] as [number, number] Defined in api/sizes.ts:31 C6 • C6: [number, number] = [323.15, 459.21] as [number, number] Defined in api/sizes.ts:32 C7 • C7: [number, number] = [229.61, 323.15] as [number, number] Defined in api/sizes.ts:33 C8 • C8: [number, number] = [161.57, 229.61] as [number, number] Defined in api/sizes.ts:34 C9 • C9: [number, number] = [113.39, 161.57] as [number, number] Defined in api/sizes.ts:35 Executive • Executive: [number, number] = [521.86, 756.0] as [number, number] Defined in api/sizes.ts:47 Folio • Folio: [number, number] = [612.0, 936.0] as [number, number] Defined in api/sizes.ts:48 Legal • Legal: [number, number] = [612.0, 1008.0] as [number, number] Defined in api/sizes.ts:49 Letter • Letter: [number, number] = [612.0, 792.0] as [number, number] Defined in api/sizes.ts:50 RA0 • RA0: [number, number] = [2437.8, 3458.27] as [number, number] Defined in api/sizes.ts:37 RA1 • RA1: [number, number] = [1729.13, 2437.8] as [number, number] Defined in api/sizes.ts:38 RA2 • RA2: [number, number] = [1218.9, 1729.13] as [number, number] Defined in api/sizes.ts:39 RA3 • RA3: [number, number] = [864.57, 1218.9] as [number, number] Defined in api/sizes.ts:40 RA4 • RA4: [number, number] = [609.45, 864.57] as [number, number] Defined in api/sizes.ts:41 SRA0 • SRA0: [number, number] = [2551.18, 3628.35] as [number, number] Defined in api/sizes.ts:42 SRA1 • SRA1: [number, number] = [1814.17, 2551.18] as [number, number] Defined in api/sizes.ts:43 SRA2 • SRA2: [number, number] = [1275.59, 1814.17] as [number, number] Defined in api/sizes.ts:44 SRA3 • SRA3: [number, number] = [907.09, 1275.59] as [number, number] Defined in api/sizes.ts:45 SRA4 • SRA4: [number, number] = [637.8, 907.09] as [number, number] Defined in api/sizes.ts:46 Tabloid • Tabloid: [number, number] = [792.0, 1224.0] as [number, number] Defined in api/sizes.ts:51 It shouldn't take long for Javascript to be able to render PDF documents. Meanwhile, the somewhat opposite functionality - executing Javascript in PDF documents - is available for quite a long time already. This article is about this functionality. Any software contains not only an actively used set of features but also a share of rarely used features. Sometimes latter set can be significant in size. You can think of features of Microsoft Word or your favorite IDE, for example. Most probably, there are enough features in them that you never used. Portable Document Format is also has many rarely used features. We are all accustomed to the text and images in PDF documents, but this is only a subset of what the format offers. PDF has set of features for creating documents that could change their contents in response to reader actions. One of such features is the ability to use Javascript in PDF documents. Javascript in PDF is most often used for the following tasks: To change document contents in response to some events. For example, to hide part of document before printing or pre-fill some form fields when document is opened. To restrict actions of the reader. For example, to validate entered form field values. To introduce malicious code in documents. A Javascript API was developed for PDF viewers to be able to interpret Javascript code. The API was primarily developed for Adobe Acrobat family of products but alternative viewers usually also support a subset of the API. Let's take a look at some samples. Hello World Here is a traditional "Hello World" sample for a start. Please note that I'll use C# and Docotic.Pdf library for the samples. You can download source code of all the samples at the end of the article. using BitMiracle.Docotic.Pdf; namespace JavascriptInPdf { public static class Demo { public static void Main(string[] args) { PdfDocument pdf = new PdfDocument(); pdf.OnOpenDocument = pdf.CreateJavaScriptAction("app.alert(\"Hello CodeProject!\", 3);"); pdf.Save("Hello world.pdf"); } } } You should see something like on the following screenshot if you open the PDF created by the sample in Adobe Reader: So, what's done by the sample? Here is the essential line: pdf.OnOpenDocument = pdf.CreateJavaScriptAction("app.alert(\"Hello CodeProject!\", 3);"); PDF supports actions. It is something that is done when an event occurs. For example, clicking on a link in a PDF document outline might trigger an action that cause viewer to open a page in the document: There are different types of actions defined. One of the types is Javascript actions. An action of this type can be created using PdfDocument.CreateJavaScriptAction method. The method accepts Javascript code. Then the action can be attached to the OnOpenDocument event. Obviously, the event occurs when document is opened in a viewer. Here is the Javascript code: app.alert("Hello CodeProject!", 3); Static class app is part of the Javascript API. This class provides methods for communicating with a PDF viewer. In particular, this class contains alert method with a number of overloads. The alert method can be used to present a modal window with a message. The code above uses the overload with optional second parameter. This parameter specifies an icon to be displayed (3 is a code for Status Icon). More complex scenarios Let's try something more pragmatic. There are many PDF documents with fillable forms. A document with a form can be a deposit account agreement, a visa application form, a questionnaire etc. Such documents can be conveniently filled right in a viewer and printed or saved for later use. Creators of such documents can employ Javascript to farther improve user experience. Fillable forms often contain date fields. These fields might look like this: Sure, a creator of a document can just add date field to the document and stop there. However, it would be nice to assist someone who will fill document later by putting current date to the date field. In most cases this will be just what is expected. Default value for a date field I won't go into details of how to create a date field. Let's focus on the Javascript part only. So, we need to put current date into a date field (there is three fields actually, one for day, month and year). Here is the script: function setDay(date) { var dayField = this.getField("day"); if (dayField.value.length == 0) { dayField.value = util.printd("dd", date); } } function setMonth(date) { var monthField = this.getField("month"); if (monthField.value.length == 0) { monthField.value = util.printd("date(en){MMMM}", date, true); } } function setYear(date) { var yearField = this.getField("year"); if (yearField.value.length == 0) { yearField.value = util.printd("yyyy", date); } } function setCurrentDate() { var now = new Date(); setDay(now); setMonth(now); setYear(now); } setCurrentDate(); As you can see, there is much more code than in “Hello World” sample. Using a string for all this code is probably not very convenient because of need to escape quotes. And editing such code later might be painful because of lack of formatting. Let's put a text file with code into application resources and use another overload of CreateJavaScriptAction method. pdf.OnOpenDocument = pdf.CreateJavaScriptAction(Resources.SetCurrentDate); When opened, a document with the script should look like the following: Take a closer look at following part of the code: monthField.value = util.printd("date(en){MMMM}", date, true); Please notice that I am using an overload of util.printd method to get localized name of the month. This works as expected in Adobe Reader but, unfortunately, other PDF viewers might not support all built-in methods of Javascript API. This should be taken into account if you are going to support anything other than Adobe Reader. Another approach is to implement custom code that does the same as util.printd. Please also notice that code populates a field only if it's empty. Without this additional check a following scenario might occur: a person fills the document, saves it but next time the document is opened all date fields are reset with current date. Validation of data Suppose you want to ensure that day and year are numbers. It's easy. Let's use following code for this: function validateNumeric(event) { var validCharacters = "0123456789"; for (var i = 0; i < event.change.length; i++) { if (validCharacters.indexOf(event.change.charAt(i)) == -1) { app.beep(0); event.rc = false; break; } } } validateNumeric(event); PDF fields fire OnKeyPress event whenever something is typed in a field. You might want to create an action with the validation code and attach it to OnKeyPress event of fields. PdfJavaScriptAction validateNumericAction = m_document.CreateJavaScriptAction(Resources.ValidateNumeric); dayTextBox.OnKeyPress = validateNumericAction; yearTextBox.OnKeyPress = validateNumericAction; After that a person filling the form won't be able to put anything other than numbers in day and year fields. Anything getting pasted from clipboard will also be validated. Synchronization of data Quite often same data should be put several times in different parts of a form. In such cases some Javascript code can help to eliminate the need for this repetitive efforts. Assume that there is a document like this: It would be nice to have both fields synchronized. It's possible with following Javascript method: function synchronizeFields(sourceFieldName, destinationFieldName) { var source = this.getField(sourceFieldName); var destination = this.getField(destinationFieldName); if (source != null && destination != null) { destination.value = source.value; } } The Javascript method could be used like this: PdfDocument pdf = new PdfDocument(“Names.pdf”); pdf.SharedScripts.Add(pdf.CreateJavaScriptAction(Resources.SynchronizeFields)); pdf.GetControl("name0").OnLostFocus = pdf.CreateJavaScriptAction("synchronizeFields(\"name0\", \"name1\");"); pdf.GetControl("name1").OnLostFocus = pdf.CreateJavaScriptAction("synchronizeFields(\"name1\", \"name0\");"); pdf.Save("NamesModified.pdf"); This will ensure that data in both fields is the same at all times. The data is synchronized whenever any field looses focus. Please notice that synchronizeFields method is put is shared scripts collection (PdfDocument.SharedScripts). This is done to use the same code from several actions. Summary As you can see, Javascript can be used not only in web development. With some additional efforts a PDF form with Javascript code can be created. And such form could please those who will fill it almost like an application with a well thought-out UI. Don't get me wrong, most important part of a PDF file is its content. Javascript is just a nice addition. And there are some imitations: Writing Javascript code for PDF files is not as convenient as for web pages There is incomplete support for Javascript API in PDF viewers other than Adobe Reader Execution of scripts can be disabled in a viewer Also, Javascript in PDF files is a security threat. From time to time different vulnerabilities are found (and fixed) in viewers. Knowing all that don't be surprised if an opened PDF file will offer you a chess game to distract you from malicious acts getting performed in background. " /> Download sample code Download javascriptinpdf.zip - 1.3 MB