

BlitzPoint 1.0

Einführung:

BlitzPoint ist ein Programm, mit dem man Präsentationen erstellen kann, ähnlich den großen Vorbildern PowerPoint für Windows, Key-Note für Apple und OpenOffice-Impress.

Da BlitzPoint autark ist und für alle drei Systemplattformen zur Verfügung stehen wird, hat man die Gewissheit, dass die vorbereiteten Präsentationen **wirklich** überall funktionieren und identisch aussehen, was selbst bei OpenOffice nach meiner Erfahrung leider nicht immer der Fall ist.

Wenngleich BlitzPoint im Umfang selbstverständlich nicht mit den professionellen Vorbildern gleichzusetzen ist, so lehnt es sich doch an deren Funktionen an, übernimmt einige ihrer Vorzüge und bietet manche Extras. Doch BlitzPoint befindet sich in einem sehr frühen Entwicklungsstand und wird noch einige Funktionserweiterungen erfahren (müssen).

Bisher läuft die Erstellung der Präsentationen für BlitzPoint über ein Skript, dessen Syntax in der folgenden Befehlsreferenz aufgelistet wird. Dieser Skript muss als Datei mit der Endung **.bps** (für **BlitzPointScript**) gespeichert werden. Kompiliert bzw. ausgeführt wird der Skript, indem man das Programm (z.B. per Konsole) mit ihr als Startargument öffnet. Unter Windows geschieht das ganz einfach, indem man die Skript-Datei auf die exe-Datei zieht. Ich habe mir erlaubt, die Syntax in deutscher Sprache zu verfassen, und habe mich auch nicht gescheut, deutsche Sonderbuchstaben zu verwenden. Daher müssen Nutzer von Unix-basierten Betriebssystemen wie Linux oder MacOS darauf achten, dass ihre Skript-Dateien **nicht** UTF-8 codiert sind, sondern nach westlichem ISO-Standard, wie es bei Windows üblich ist.

Zukünftig geplant ist die Entwicklung einer integrierten Entwicklungsumgebung, mit der sich die Skripte schneller, leichter, anschaulicher und ohne Risiko von Schreibfehlern erstellen lassen.

Die Skript-Methode hat ihren Grund im BCC-27. Um daran teilnehmen zu können, musste ich neben der Vollversion noch eine reduzierte Fassung anbieten: Der Code-Umfang sprengte nämlich den vorgeschriebenen Rahmen von 28 Kilobyte und enthielt einige Funktionen und Klassen, die ich schon vor dem BCC erstellt hatte. Um das Soll zu erfüllen, habe ich alles Verbotene 'rausgeschmissen und das Code-Korsett auf 26 Kilobyte zusammengequetscht im wahrsten Sinne des Wortes. Der Funktionsumfang ist jedoch beinahe derselbe wie in der Vollversion. Einzig und allein die sehr aufwendig programmierten animierten Seitenwechsel in 35 Varianten(!) musste ich für den BCC opfern. Vielleicht könnte sich trotzdem der ein oder andere mit der Vollversion einen Eindruck der Seitenwechselanimationen verschaffen.

Der reduzierte Code ist zum Lesen ungenießbar. Wenn jemand sich **wirklich** die Mühe machen will, den Code zu studieren, empfehle ich ihm unbedingt, die strukturierten, in einzelne Klassen ausgelagerten und kommentierten Codes der Vollversion zu lesen. Alles was mit der Klasse TPageChange zu tun hat, kann bezüglich des BCC's ignoriert werden, da sie für die Seitenwechselanimationen zuständig ist.

Befehlsreferenz:

Man darf im Skript eigentlich so gut wie alles hineinschreiben, was man möchte. Kommentare müssen nicht extra gekennzeichnet oder abgesetzt werden. **Einzig und allein das Zeichen # am Anfang einer Zeile** signalisiert dem BlitzPoint-Compiler, dass es sich um ein Schlüsselwort handeln könnte. Tabulator-Eintrückungen vor dem # werden ignoriert. Die unten aufgelisteten Schlüsselwörter müssen in der richtigen Groß- und Kleinschreibung notiert sein. Verlangen die Schlüsselwörter ein nachfolgendes Argument, muss dieses (ohne zusätzliche Leerzeichen!) hinter einem direkt angefügten Gleichheitszeichen = folgen.

Syntax:	Argument-Beispiele:	Erläuterung:
#Element=	Willkommen bei BlitzPoint media/images/picture.png	Dies ist der grundlegende Befehl von BlitzPoint. Mit ihm wird der (aktuellen) Präsentations-Seite ein Text oder Bild hinzugefügt. Entweder ist das Argument ein gültiger Pfad zu einer Bilddatei. Ansonsten wird es von BlitzPoint als Textzeile interpretiert.

Anmerkungen:

- Im momentanen Entwicklungsstand werden alle Elemente einer Seite automatisch untereinander angeordnet bzw. formatiert. In einer zukünftigen Version wird auch das Nebeneinander der Elemente möglich sein.
- Bezüglich eines hinzuzufügenden Elements können einige Einstellungen bestimmt werden. Sie gelten **global** ab jener Stelle im Skriptverlauf, in der sie eingefügt wurden. Will man also ein Element beeinflussen, müssen **vorher** entsprechende Befehle erfolgen. Will man sie nicht für die weiter folgenden Elemente anwenden, müssen sie nach dem Hinzufügen des Elements wieder rückgängig gemacht werden.

Syntax:	Argument-Beispiel	Erläuterung:
#Schriftgröße=	24	(Bezieht sich wie das Folgende natürlich nur auf Text-Elemente)
#Schriftart=	media/fonts/fontin.ttf	Es ist die kostenlose Schriftart "fontin" von Jos Buivenga beigelegt, auf die BlitzPoint standardmäßig zugreift. Aber es kann auch eine andere Schriftartdatei (mit richtigem Pfad) geladen werden
#Ausrichtung=	Links Rechts Zentriert	Die Argumente beschränken sich auf diese drei Angaben. Die horizontale Anordnung ist nicht absolut, sondern richtet sich relativ nach den Rändern des vorgehenden Text- oder Bild-Elements und muss nach Bedarf noch mit dem nächsten Befehl korrigiert werden
#Anordnung=	<i>Untereinander Nebeneinander</i>	<i>Dieser Befehl bleibt zukünftigen BlitzPoint-Versionen vorbehalten</i>
#Einrücken=	50	Alles wird global um (50) Pixel nach Rechts gerückt. Mit negativen Werten erfolgt das Rücken nach Links. Hier muss ggf. ein bisschen experimentiert werden.
#Farbe=	Weiß Hellgrau Grau Dunkelgrau Schwarz Hellrot Rot Dunkelrot Hellgrün Grün Dunkelgrün Hellblau Blau Dunkelblau Hellgelb Gelb Dunkelgelb Helltürkis Türkis Dunkeltürkis Hellviolett Violett Dunkelviolet Hellorange Orange Hellbraun Braun	Diese Argumente sind festgelegt und können nicht mehr weiter differenziert werden. Sie gelten für alle dem Befehl folgende Elemente.
#Dauer=	2000 14400000 -1	Dieser wichtige Befehl legt die Fortsetzung der Elemente auf einer Seite fest. Übergeben wird die Zeitdauer in Millisekunden, die angibt, wie lang das Element verharrt, bis das nächste eintritt. 2000 bedeutet also 2 Sekunden. Möchte man, dass die Präsentation ganz "einfriert", übergibt man einfach einen sehr hohen Integer-Wert von z.B. 14400000, was vier Stunden bedeutet! (So lange wird keiner warten ...) Will man hingegen, dass die Präsentation erst dann fortgesetzt wird, wenn der Benutzer die Enter-Taste, die Leertaste oder die Pfeiltaste nach Rechts oder Unten tippt, dann signalisiert man das mit dem Argument -1

Syntax:	Argument-Beispiel	Erläuterung:
#Sound=	media/sounds/klang.ogg	Das Argument bildet hier ein Pfad zu einer gültigen Sound-Datei. Diese wird geladen und abgespielt, sobald das Element auf der Seite steht. Will man kein Klang mehr, übergibt man einfach nichts: #Sound=
#Animation=	Einblenden Zoomen-A Zoomen-B Falten Verschieben von Links Verschieben von Rechts Verschieben von Unten Verschieben von Oben Aufziehen von Links Aufziehen von Rechts Aufziehen von Oben Aufziehen von Unten Mosaik Drehen Zittern Pseudo-3D-A Pseudo-3D-B Zufällig	Diese 18 Argumente sind festgelegt. Mit ihnen werden die Elemente auf verschiedene Arten in die Seite hineinbewegt. Erst wenn die Animation fertig ist, kommen die Befehle #Sound= und #Dauer= usw. zum Tragen. Das letzte Argument <i>Zufällig</i> wählt pro Element zufällig eines der vorangehenden 17 Animationen aus.
#Animationszeit=	1000 0	Wiederum in Millisekunden wird angegeben, wie lang die Animation dauern darf, bevor das Element vollständig erschienen ist. Und wie eben angemerkt, bleibt die an die Animation anschließende Verweildauer #Dauer= davon unberührt. Will man gar keine Animation (mehr) haben, setzt man die Animationszeit auf 0
#Animationssound=	media/sounds/klang.ogg	Wiederum wird der Pfad zu einer gültigen Sounddatei übergeben. Im Gegensatz zum Befehl #Sound= erklingt der Sound aber schon zu Beginn der Animation und nicht erst, wenn das Element bereits fest auf seinem Platz ruht.
<i>Mit den folgenden Befehlen werden (beliebige) Elemente interaktiv, d.h. sie reagieren auf Mausklicks des Benutzers, zu diesem Zweck wird es nötig sein, #Dauer=1440000 zu setzen, um dem Benutzer beliebig Zeit zu geben.</i>		
#Interaktion: Gehe zu Seite=	3	Dieses Ereignis entspricht dem Goto-Befehl, mit dem man beliebig zwischen den einzelnen Seiten der Präsentation navigieren bzw. hin- und her klicken kann. Die Seitennummer ergibt sich aus der Reihenfolge im Skript.
#Interaktion: Zeige Info=	Richtig geraten	Ähnlich einer Dialogbox kann hier eine kurze Nachricht zentriert eingeblendet werden. Auf System-Dialogboxen wurde verzichtet, da sie im Vollbildmodus nicht funktionieren.
#Interaktion: Zeige Bild=	media/images/bild.jpg	Wie vorher kann ein Bild zentriert eingeblendet werden. Das Argument muss ein <u>gültiger</u> Pfad einer Bilddatei sein.
#Interaktion: Spiele Sound=	media/sounds/klang.ogg	Das Argument muss ein Pfad für eine Sounddatei sein ...
#Interaktion: Starte Skript=	anderer-Script.bps	Das Argument muss ein Pfad für eine Skript-Datei sein. Dann wird erneut BlitzPoint mit diesem Skript gestartet, so dass sich mehrere Präsentationen interaktiv verbinden lassen.
#Interaktion: Öffne URL=	/media/movie/Film.mpg	Eine andere Datei wird extern gestartet.
#Interaktion: Ende		BlitzPoint wird (ohne Argument!) beendet.
#Interaktion:		Hiermit wird die Interaktionsfähigkeit für weitere hinzuzufügende Elemente neutralisiert bzw. außer Kraft gesetzt.

Mit der folgenden Syntax lassen sich als Elemente neben Textzeilen und Bildern auch Nonstop-Animationen im Stil des Gif-Formats hinzufügen. Zusätzlich können Sounds integriert werden. Voraussetzung ist, dass (**nach** den Gif-Daten) mit *#Element=* ein gültiger Dateipfad übergeben wird, aus dem dann die Bilderframes geladen werden können.

Syntax:	Argument-Beispiel	Erläuterung:
#Gif-Anzahl=	10	Anzahl der Frames
#Gif-Breite=	128	Breite eines Frames
#Gif-Höhe=	64	Höhe eines Frames
#Gif-Sound=	media/sounds/Gif.ogg	Sounddatei, die parallel zu den Animationsdurchläufen in einer Nonstop-Schleife abgespielt wird.
#Gif-Millisekunden=	4000, 1000, 200, 80	Beginnend mit dem 1.Frame des Gif-Elements kann hier die Geschwindigkeit (in Millisekunden) bestimmt werden, mit der die einzelnen Frames ablaufen. Die Integer-Werte werden durch Kommas getrennt. Die Anzahl der übergebenen Argumente muss nicht mit der Anzahl der Frames übereinstimmen, die per <i>#Gif-Anzahl=</i> vereinbart werden. Im vorliegenden Beispiel wird für den 5. bis 10.Frame einfach die Vorgabe für den 4.Frame (80 Millisekunden) fortgesetzt.
Wie sonst auch, ist hier keinesfalls zu vergessen, die <i>#Gif-Anzahl=</i> nach dem Hinzufügen des Gif-Elements wieder auf 0 zu setzen, falls (erwartungsgemäß) die nachfolgenden Elemente <u>keine</u> Animationen mehr sein sollen! Ansonsten folgt ein Absturz, weil BlitzPoint versuchen würde, aus den folgenden Element-Argumenten animierte Bilder zu laden.		

Alle bisherigen Schlüsselwörter bezogen sich auf Elemente einer Seite. Wie aber lässt sich eine neue Seite initialisieren? Dafür gibt es die folgenden drei Möglichkeiten bzw. Schlüsselwörter:

Syntax:	Argument-Beispiel:	Erläuterung:
#Nächste Seite		Hier wird einfach eine neue Seite erstellt, die in der Chronologie der Präsentation als nächstes kommt. Ein Argument ist nicht nötig bzw. auch nicht erlaubt.
#Gehe zu Seite=	4	Nach Ablauf der aktuellen Seite wird zu einer anderen Seite "gesprungen", deren Nummer sich aus der Reihenfolge im Skript ergibt.
#Ende		Nach Ablauf der aktuellen Seite wird BlitzPoint beendet. Ein Argument ist nicht nötig bzw. auch nicht erlaubt. Der Befehl ist deswegen nötig, weil er es erlaubt, weitere Seiten im Skript anschließen zu lassen, die wie bei isolierten Unterprogrammen nur per "Sprung" erreicht werden können.

Auch einige allgemeine Einstellungen lassen sich vornehmen. Diese lassen sich nur einmal vornehmen, gelten für die gesamte Präsentation und müssen irgendwo **am Anfang des Skripts stehen** und spätestens vor dem Hinzufügen des allerersten Elements abgeschlossen sein.

Syntax:	Argument-Beispiel:	Erläuterung:
#Bildschirm=	1024x768	Getrennt durch ein "x" wird die Bildschirmauflösung bestimmt. Standardmäßig sind 800 x 600 Pixel eingestellt.
#Modus=	<i>Fenster Vollbild</i>	<i>Dieser Befehl bleibt zukünftigen BlitzPoint-Versionen vorbehalten. Bisher ist nur der Fenstermodus möglich.</i>
#Hintergrunddatei=	media/images/background.jpg	Das Argument muss ein gültiger Pfad einer Bilddatei mit den richtigen Maßen sein, welche als Hintergrund für die gesamte Präsentation dient
#Export=	Präsentation	Sobald dieses Schlüsselwort im Skript auftaucht, wird die Präsentation nicht vorgeführt. Stattdessen werden die einzelnen Seiten komplett als png-Bilder exportiert. Mit dem Argument kann ein vereinbar Dateiname definiert werden.

Mit dem "*Demo-Skript.bps*" können Sie sich einen ersten Eindruck der oben geschilderten Möglichkeiten machen.

Es folgen nun jene Befehle, die aus oben genannten Gründen nur in der Vollversion enthalten sind, in der reduzierten BCC-Version aber ignoriert werden. Mit ihnen lassen sich die Wechsel der Präsentationsseiten auf 35 verschiedene Arten animieren. Dieses Feature lässt BlitzPoint auch für eine unterhaltsame Fotoshow in Frage kommen. Interessierte testen bitte den Skript *"Slideshow.bps"* (mit der Vollversion).

Syntax:	Argument-Beispiel:	Erläuterung:
#Seitenwechselzeit=	2000	Das Argument (in Millisekunden) bestimmt, wie lang die Animation des Seitenwechsel dauern soll.
#Seitenwechselsound=	media/images/klang.ogg	Das Argument muss ein gültiger Pfad einer Sounddatei sein, die mit dem Seitenwechsel abgespielt wird.
#Seitenwechsel=	Überblenden Spotlight Drehscheibe abwärts Drehscheibe aufwärts Schleuse Verschieben nach Links Verschieben nach Rechts Verschieben und Zoomen nach Links Verschieben und Zoomen nach Rechts Abblättern Aufblättern Kacheln zoomen Kacheln blitzen Kacheln rotieren von Rechts Kacheln rotieren von Oben Kachelpermutation Kacheln verdrehen Kacheln fallen Kippen 3D-Seite-A 3D-Seite-B 3D-Blatt-A 3D-Blatt-B 3D-Türen-A 3D-Türen-B 3D-Türen-C 3D-Türen-D 3D-Quader dreht nach Unten 3D-Quader dreht nach Oben 3D-Quader dreht nach Rechts 3D-Quader dreht nach Links 3D-Ecke dreht nach Unten 3D-Ecke dreht nach Oben 3D-Ecke dreht nach Rechts 3D-Ecke dreht nach Links Zufall-Einfach Zufall-Verschieben Zufall-Kacheln Zufall-2D Zufall-3D Zufall-Spezial Zufall	<p>Die Beschreibung der Animationen sollten selbsterklärend sein. Ansonsten gilt: Einfach ausprobieren!</p> <p>Die zuletzt aufgeführten Zufallsvarianten bedeuten, dass nur aus einer Auswahl der 35 Möglichkeiten zufällig ausgewählt wird:</p> <p>(Von "Überblenden" bis "Schleuse") (Die vier "Verschieben"-Varianten) (Die sieben "Kachel"-Varianten) (Von "Überblenden" bis "Kacheln fallen") (Von "Kippen" bis zu den "3D-Ecken...") (Meine persönlichen zehn Favoriten...) Total zufällig</p>