

Palette

COLLABORATORS

	<i>TITLE :</i> Palette		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		January 19, 2025	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	Palette	1
1.1	Palette V1.10	1
1.2	blue	1
1.3	createpalette	2
1.4	displaypalette	2
1.5	fade	2
1.6	fadeout	3
1.7	freepalette	3
1.8	getpicturepalette	3
1.9	getscreenpalette	4
1.10	green	4
1.11	initpalette	4
1.12	nbcolour	4
1.13	palrgb	4
1.14	red	5
1.15	rgb	5
1.16	usepalette	5

Chapter 1

Palette

1.1 Palette V1.10

Pure Basic Palette library V1.00

Paletten sind für alle angezeigten Elemente sehr wichtig. Sie werden sehr gut unterstützt und Sie können fast alles damit anstellen. Spezielle Funktionen wie Fading Routinen ermöglichen Ihren Programmen sehr gute Effekte.

Befehlsübersicht:

- Blue
- CreatePalette
- DisplayPalette
- Fade
- FadeOut
- FreePalette
- GetPicturePalette
- GetScreenPalette
- Green
- InitPalette
- NbColour
- PalRrgb
- Red
- Rgb
- UsePalette

Beispiel:

Fading Workbench

1.2 blue

SYNTAX

Blue.w = Blue(ColourIndex)

FUNCTION

Gibt den Blau-Wert der in der aktuellen Palette gefundenen Farbe zurück. Der zurückgegebene Wert liegt immer zwischen 0 und 255.

1.3 createpalette

SYNTAX

```
res.l = CreatePalette(#Palette, NbColour)
```

COMMAND

Versucht mit den angegebenen Argumenten eine neue Palette zu erstellen. Die Größe, die eine solche Palette im Speicher belegt, kann wie folgt berechnet werden:

Größe (in Bytes) = NbColours * 12 + 12

Die erstellte Palette ist benutzungsfertig und gefüllt mit der Farbe 0.

1.4 displaypalette

SYNTAX

```
DisplayPalette(#Palette, ScreenID)
```

STATEMENT

Zeigt die angegebene #Palette auf dem Bildschirm an.

1.5 fade

SYNTAX

```
Fade(#Palette1, #Palette2, Step, NbLoop, ScreenID)
```

STATEMENT

Vollführt einen hübschen Umblendprozeß (Fade) zwischen zwei Paletten. Die Paletten müssen die selbe Farbanzahl aufweisen oder es kann zu einem Absturz kommen.

'Step' bestimmt die Geschwindigkeit des Fade-Prozesses (1 ist die schnellste, Werte >1 verlangsamen die Geschwindigkeit).

'NbLoop' bestimmt, wieviele Schleifen der Fading-Prozeß bis zum Ende durchlaufen muß. Standardmäßig werden IMMER 255 Schleifen ausgeführt. Sie können diesen Wert hiermit manuell einstellen (z.B. mit einem 'Step' Wert von 2, sollten Sie einen NbLoop Wert von 255/2 += 127 verwenden).

Diese Funktion ist auf Geschwindigkeit optimiert und ergibt sehr gute Resultate auf allen Amigas (68020 wird trotzdem empfohlen), auch auf vielfarbigen Bildschirmen (bis zu 256 Farben).

1.6 fadeout

SYNTAX

```
FadeOut(#Palette, Step, NbLoop, ScreenID)
```

STATEMENT

Stellt mit der angegebenen #Palette einen sehr hübschen Ausblend-Prozeß dar. Die Palette WIRD dabei verändert (am Ende des Fading-Prozesses ist die Palette komplett schwarz).

- . Die Ausblendgeschwindigkeit kann mit dem 'Step' Parameter eingestellt werden.

Ist Step = 1 ergibt dies ein sanftes Fading und wartet 1 VWait vorm Fading des nächsten Frames.

Ist Step = 2 ist das Fading zweimal so schnell wie mit Step 1 ...

NbLoop wird benutzt, um den Bildschirm teilweise auszublenden:

Ist NbLoop = 255 ist am Ende der gesamte Bildschirm schwarz, da mit 255 Schleifen (Loops) das Fading komplett ist.

Ist NbLoop = 50, wird FadeOut nach 50 Schleifen gestoppt. Testen Sie es zum besseren Verständnis :)

Diese Routine ist auf Geschwindigkeit optimiert und ergibt auch auf kleinen Amigas exzellente Ergebnisse. And noch mehr, sie ist absolut systemfreundlich (keine direkte Hardware Ansteuerung...) und arbeitet daher auch auf Grafikkarten! Es ist besser, diese Routine anstelle von Fade() zu benutzen, um ein standardmäßiges Ausblenden (Fade Out) zu erreichen...

1.7 freepalette

SYNTAX

```
FreePalette(#Palette)
```

STATEMENT

Gibt den von der angegebenen #Palette reservierten Speicher frei.

1.8 getpicturepalette

SYNTAX

```
res.l = GetPicturePalette(#Palette, PictureID)
```

COMMAND

Versucht, eine neue Palette - gefüllt mit den Farbinformationen des Bildes - zu erstellen.

Ist res = 0 konnte die Palette nicht erstellt werden.

PictureID ist ein Zeiger auf eine IFF/ILBM Datei im Speicher.

1.9 getscreenpalette

SYNTAX

```
res.l = GetScreenPalette(#Palette, ScreenID)
```

COMMAND

Versucht, eine neue Palette – gefüllt mit den Farbinformationen des aktuellen Bildschirms – zu erstellen.

Ist res = 0 konnte die Palette nicht erstellt werden.

1.10 green

SYNTAX

```
Green.w = Green(ColourIndex)
```

FUNCTION

Gibt den Grün-Wert der in der aktuellen Palette gefundenen Farbe zurück. Der zurückgegebene Wert liegt immer zwischen 0 und 255.

1.11 initpalette

SYNTAX

```
result.l = InitPalette(#NumPaletteMax)
```

FUNCTION

Initialisiert die gesamte Paletten-Programmierungsumgebung zur späteren Benutzung. Sie müssen diese Funktion am Anfang Ihres Sourcecodes aufrufen, wenn Sie die Paletten-Befehle benutzen möchten.

#NumPaletteMax : Maximale Zahl zu verwaltender Paletten.

1.12 nbcolour

SYNTAX

```
Result.l = NbColour
```

STATEMENT

Gibt die Anzahl Farben der gerade benutzten Palette zurück.

1.13 palrgb

SYNTAX

```
PalRgb(ColourIndex, R, G, B)
```

STATEMENT

Ändert den RGB-Wert einer Farbe in der aktuellen Palette.

1.14 red

SYNTAX

`Red.w = Red(ColourIndex)`

FUNCTION

Gibt den Rot-Wert der in der aktuellen Palette gefundenen Farbe zurück. Der zurückgegebene Wert liegt immer zwischen 0 und 255.

1.15 rgb

SYNTAX

`Rgb(ScreenID, ColourIndex, R, G, B)`

STATEMENT

Ändert direkt den RGB-Wert einer Farbe des angegebenen Bildschirms.

1.16 usepalette

SYNTAX

`UsePalette(#Palette)`

STATEMENT

Ändert die aktuelle Palette auf die angegebene #Palette.
