A I. Das VectorScript “myScriptName”

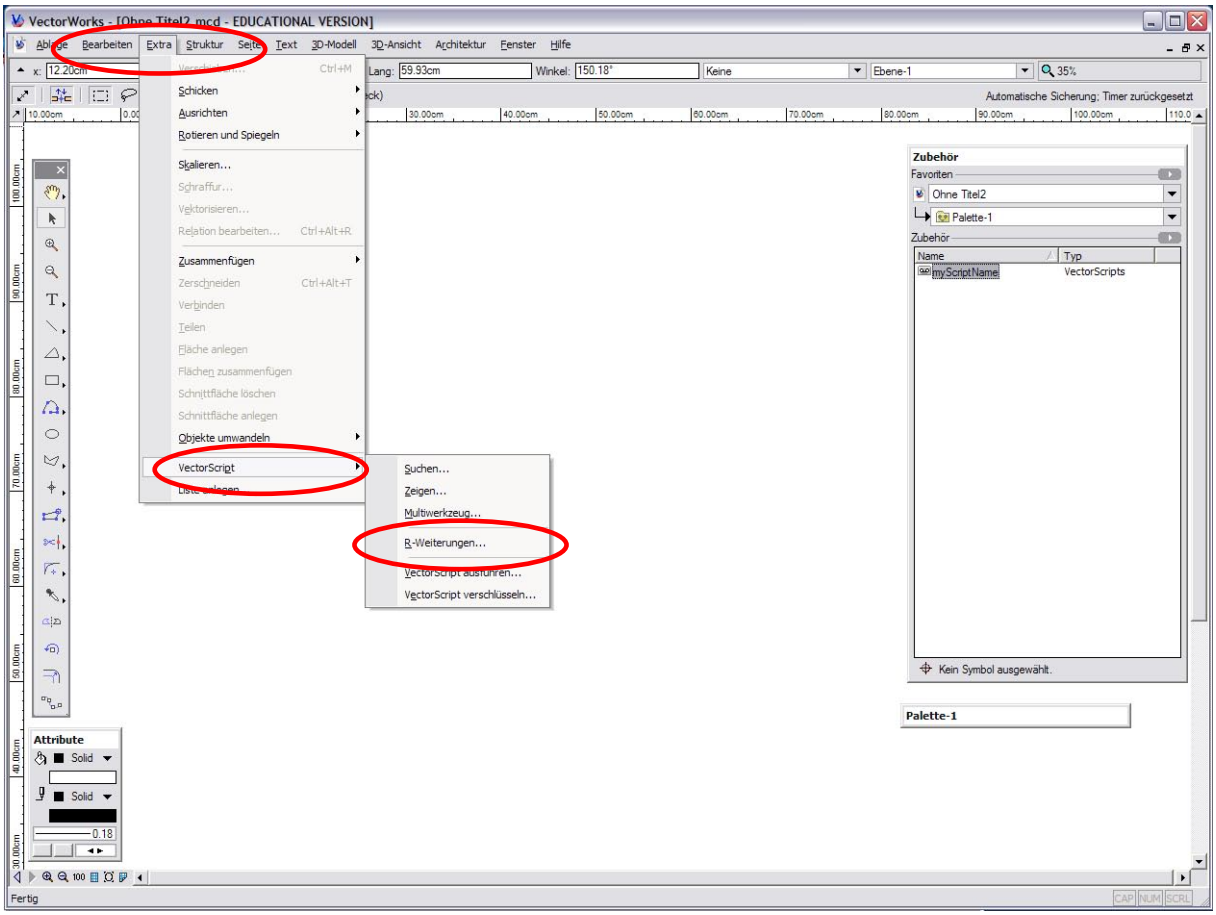
```
{Anschrift und Anrede}
PROCEDURE myScriptName;

{ dieses script soll eine bestimmte anzahl kreise geordnet in eine fläche  (rechteckig)  zeichnen}

{Einleitung}
VAR
	{	hier werden "globale" variablen aufgeführt, welche im script benutzt werden sollen. auf diese variablen kann
		auch von den subproceduren zugegriffen werden.
		sinnvoll ist die variablen geordnet aufzuführen, zum beispiel nach dem bereich in dem sie verwendet werden sollen.
		wichtig ist den variablentyp passend zum Zweck festzulegen.
	}

	{groesse der flaeche, rechtecks}
	unitsOfLength, unitsOfWidth : INTEGER;
	sizeOfUnit, radius, posX, posY : REAL;

	{Schleifen, Loops}
	i, k : INTEGER;

	{platzhalter hanswurst in drawCircleByCentreAndRadius}
	nichtbenoetigteVariable : INTEGER;

	{
		in diesem bereich werden zudem subproceduren deklariert, welche im Script verwendet werden sollen.
		subproceduren sind kleine unterprogramme, welche verwendet werden können um immer wiederkehrende
		aufgaben auszuführen.
		der aufbau ist wie schon bekannt: PROCEDURE - VAR - BEGIN - END .
	}

	{	BESCHREIBUNG DER SUBPROCEDURE}
	{	diese subprocedure soll einen Kreis zeichnen mit der Angabe eines Kreiszentrums und des Radius.
		aufbau ist folgender:
		PROCEDURE nameDerProcedure(VAR --> beschreibung der variablen, welche man beim aufruf der procedure mitgibt);
	}

	{Anschrift und Anrede der subprocedure}
	PROCEDURE drawCircleByCentreAndRadius(VAR rad, posCircleX, posCircleY : REAL; VAR hanswurst : INTEGER);
			{ hanswurst wirst nicht verwendet werden, zweck ist nur zu veranschaulichen wie verschieden variablentypen deklariert werden..}

	{Einleitung  der subprocedure }
	VAR
		{hier werden variablen beschrieben, die man nur in der subprocedure braucht! zum beispiel:}
		counter : INTEGER;

	{Haupteil  der subprocedure }
	BEGIN
		{ein kreis wird in vectorscript durch die punkte der "bounding box" beschrieben, d.h. den unteren linken punkt und den oberen
		rechten punkt eines rechtecks, welches den kreis einfasst.}

		 {	Oval( p1X :REAL; p1Y :REAL; p2X :REAL; p2Y	:REAL) ;

			Description:
			Procedure Oval creates an oval object in a VectorWorks document.

			Parameters:
			 p1	Top left coordinate of oval bounding box.
			 p2	Bottom right coordinate of oval bounding box.
		}
		oval(posCircleX-rad, posCircleY-rad, posCircleX+rad, posCircleY+rad);

	{Schluss  der subprocedure }
	END;

{Haupteil}

BEGIN
	{Beginn des eigentlichen Scripts }

	sizeOfUnit := 23;
	unitsOfLength := 4;
	unitsOfWidth := 5;

	radius := sizeOfUnit/2;
	nichtbenoetigteVariable := 5;

	FOR i := 1 TO unitsOfLength DO BEGIN
		FOR k := 1 TO unitsOfWidth DO BEGIN
			posX := k * sizeOfUnit;
			posY := i * sizeOfUnit;

			{drawCircleByCentreAndRadius(VAR rad, posCircleX, posCircleY : REAL; VAR hanswurst : INTEGER);}
			drawCircleByCentreAndRadius( radius, posX, posY, nichtbenoetigteVariable);
		END;
	END;
{Schluss}
END;
RUN(myScriptName);
```

A II. Das VectorScript “myScriptName” prepariert für das PlugIn

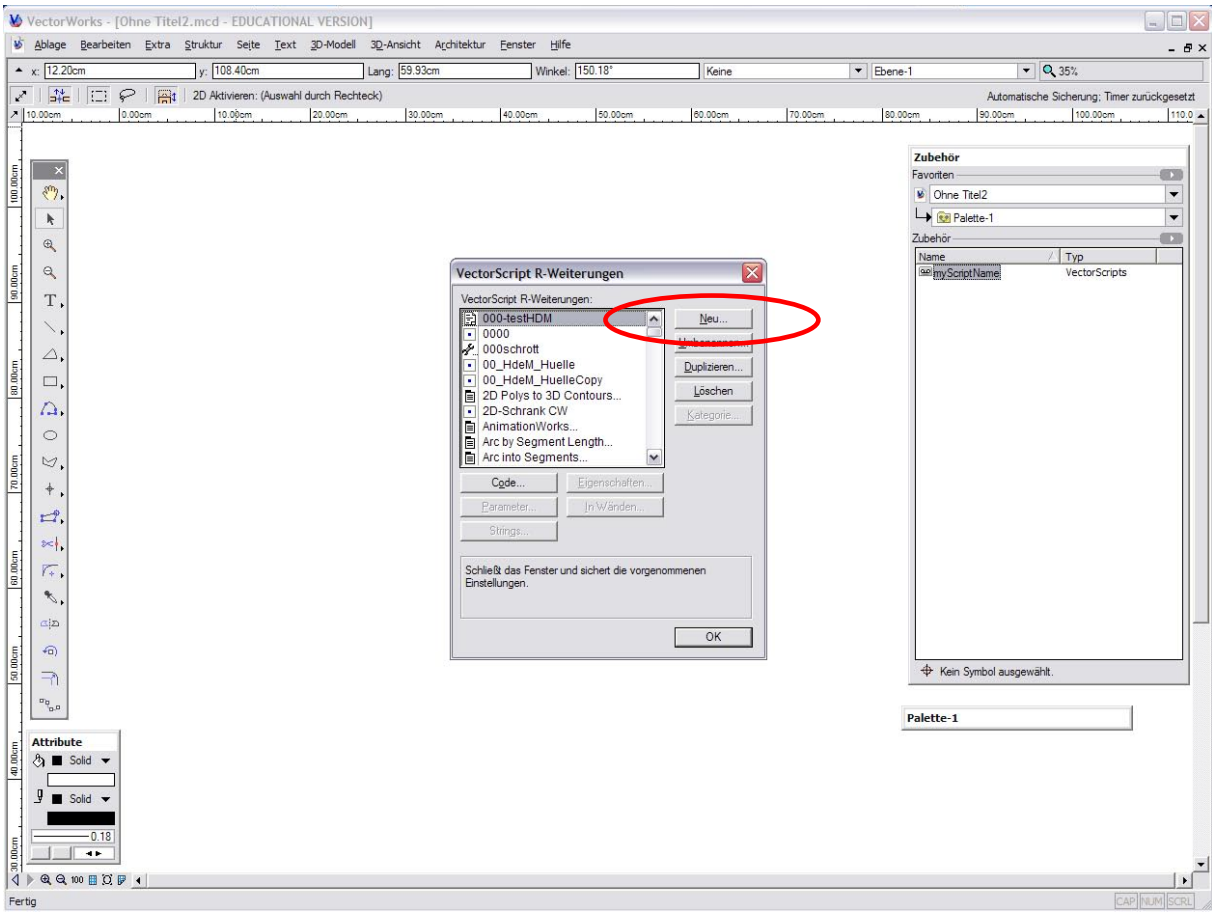
```
{Anschrift und Anrede}
PROCEDURE myScriptName;

{ dieses script soll eine bestimmte anzahl kreise geordnet in eine fläche  (rechteckig)  zeichnen}

{Einleitung}
VAR
	{	hier werden "globale" variablen aufgeführt, welche im script benutzt werden sollen. auf diese variablen kann
		auch von den subproceduren zugegriffen werden.
		sinnvoll ist die variablen geordnet aufzuführen, zum beispiel nach dem bereich in dem sie verwendet werden sollen.
		wichtig ist den variablentyp passend zum Zweck festzulegen.
	}

	{groesse der flaeche, rechtecks}
	unitsOfLength, unitsOfWidth : INTEGER;
	sizeOfUnit, radius, posX, posY : REAL;

	{Schleifen, Loops}
	i, k : INTEGER;

	{platzhalter hanswurst in drawCircleByCentreAndRadius}
	nichtbenoetigteVariable : INTEGER;

	{
		in diesem bereich werden zudem subproceduren deklariert, welche im Script verwendet werden sollen.
		subproceduren sind kleine unterprogramme, welche verwendet werden können um immer wiederkehrende
		aufgaben auszuführen.
		der aufbau ist wie schon bekannt: PROCEDURE - VAR - BEGIN - END .
	}

	{	BESCHREIBUNG DER SUBPROCEDURE}
	{	diese subprocedure soll einen Kreis zeichnen mit der Angabe eines Kreiszentrums und des Radius.
		aufbau ist folgender:
		PROCEDURE nameDerProcedure(VAR --> beschreibung der variablen, welche man beim aufruf der procedure mitgibt);
	}

	{Anschrift und Anrede der subprocedure}
	PROCEDURE drawCircleByCentreAndRadius(VAR rad, posCircleX, posCircleY : REAL; VAR hanswurst : INTEGER);
			{ hanswurst wirst nicht verwendet werden, zweck ist nur zu veranschaulichen wie verschieden variablentypen deklariert werden..}

	{Einleitung  der subprocedure }
	VAR
		{hier werden variablen beschrieben, die man nur in der subprocedure braucht! zum beispiel:}
		counter : INTEGER;

	{Haupteil  der subprocedure }
	BEGIN
		{ein kreis wird in vectorscript durch die punkte der "bounding box" beschrieben, d.h. den unteren linken punkt und den oberen
		rechten punkt eines rechtecks, welches den kreis einfasst.}

		 {	Oval( p1X :REAL; p1Y :REAL; p2X :REAL; p2Y	:REAL) ;

			Description:
			Procedure Oval creates an oval object in a VectorWorks document.

			Parameters:
			 p1	Top left coordinate of oval bounding box.
			 p2	Bottom right coordinate of oval bounding box.
		}
		oval(posCircleX-rad, posCircleY-rad, posCircleX+rad, posCircleY+rad);

	{Schluss  der subprocedure }
	END;

{Haupteil}

BEGIN
	{Beginn des eigentlichen Scripts }

	sizeOfUnit := PUNITSIZE;
	unitsOfLength := PANZAHL_X;
	unitsOfWidth := PANZAHL_Y;

	radius := PKREISRADIUS;
	nichtbenoetigteVariable := 5;

	FOR i := 1 TO unitsOfLength DO BEGIN
		FOR k := 1 TO unitsOfWidth DO BEGIN
			posX := k * sizeOfUnit;
			posY := i * sizeOfUnit;

			{drawCircleByCentreAndRadius(VAR rad, posCircleX, posCircleY : REAL; VAR hanswurst : INTEGER);}
			drawCircleByCentreAndRadius( radius, posX, posY, nichtbenoetigteVariable);
		END;
	END;
{Schluss}
END;
RUN(myScriptName);
```

B. Das PlugIn

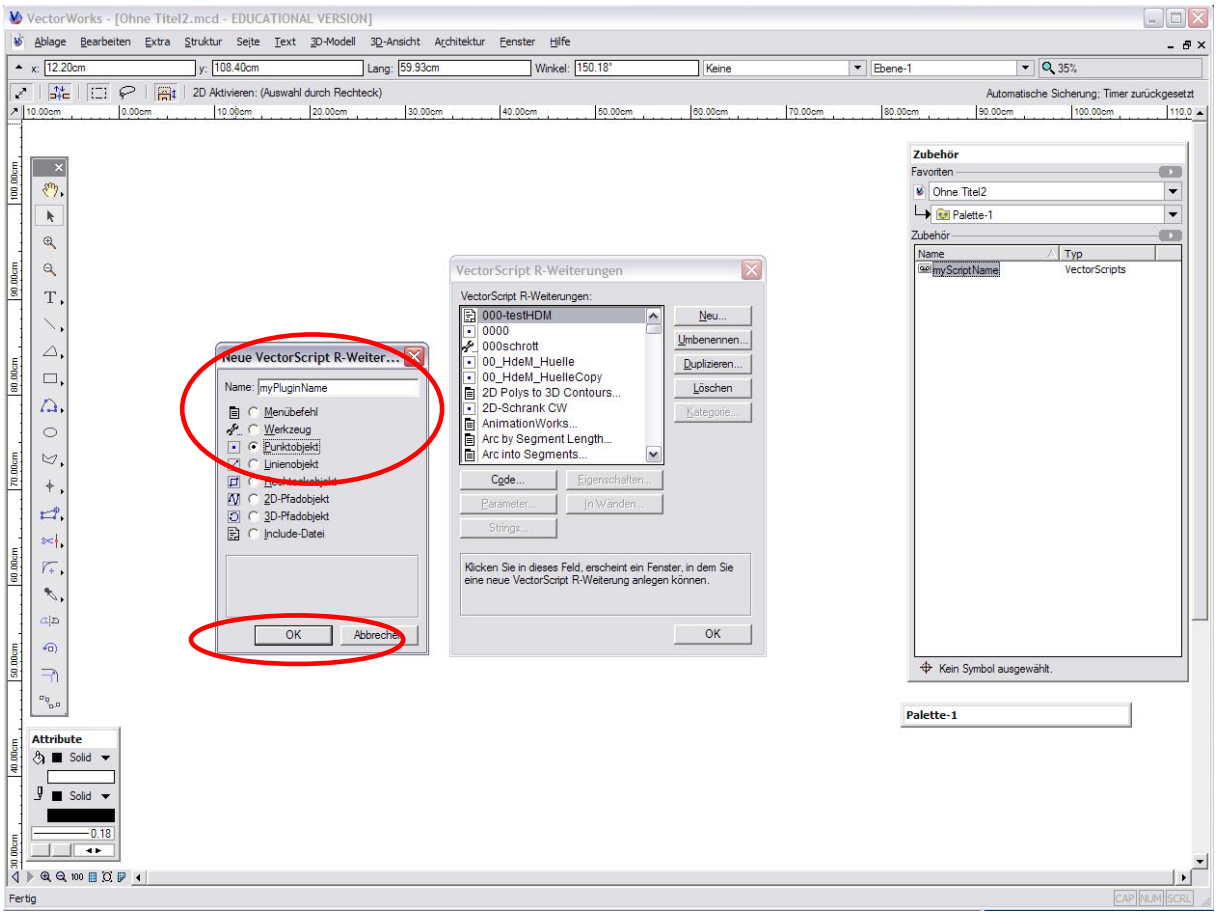
1. Das Plug In einrichten



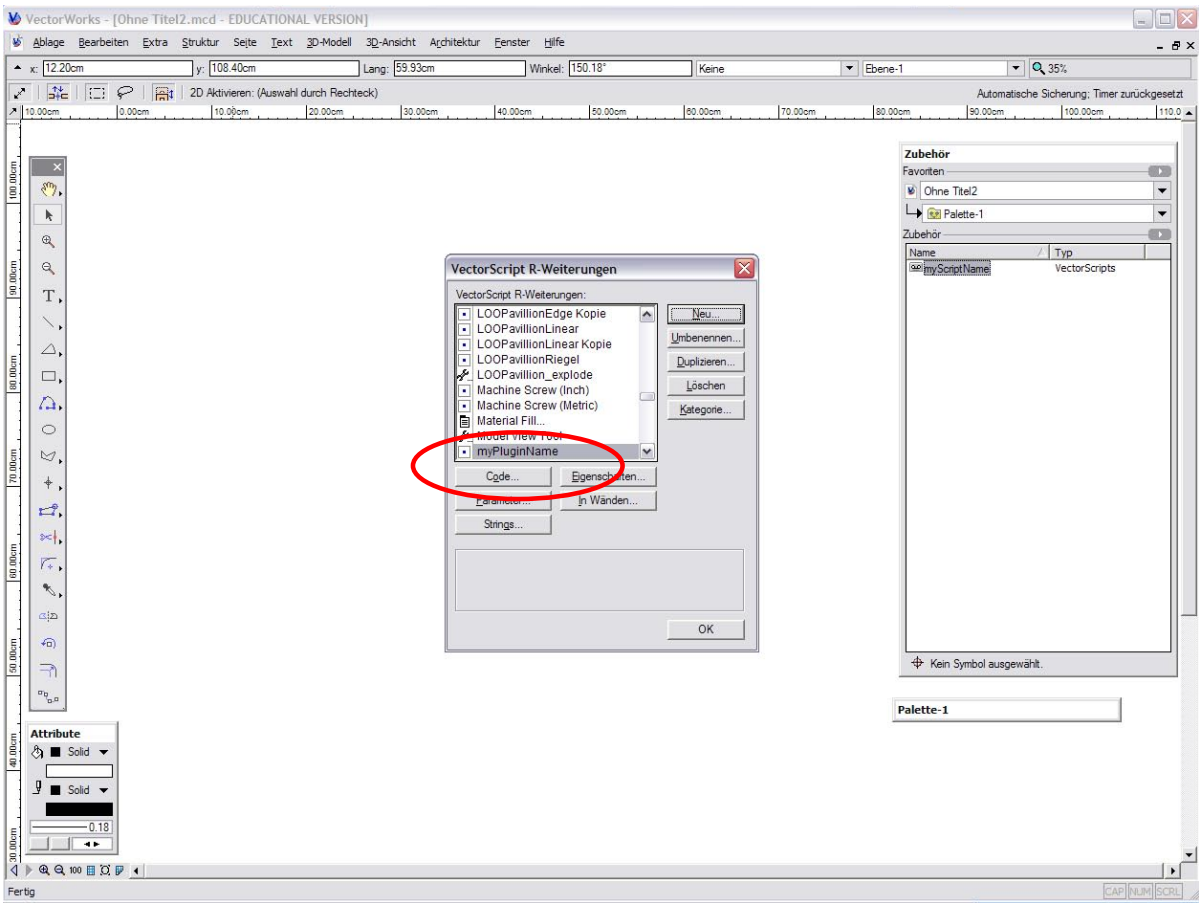
2. bestehendes Plug In wählen oder ein “NEU..”es erstellen



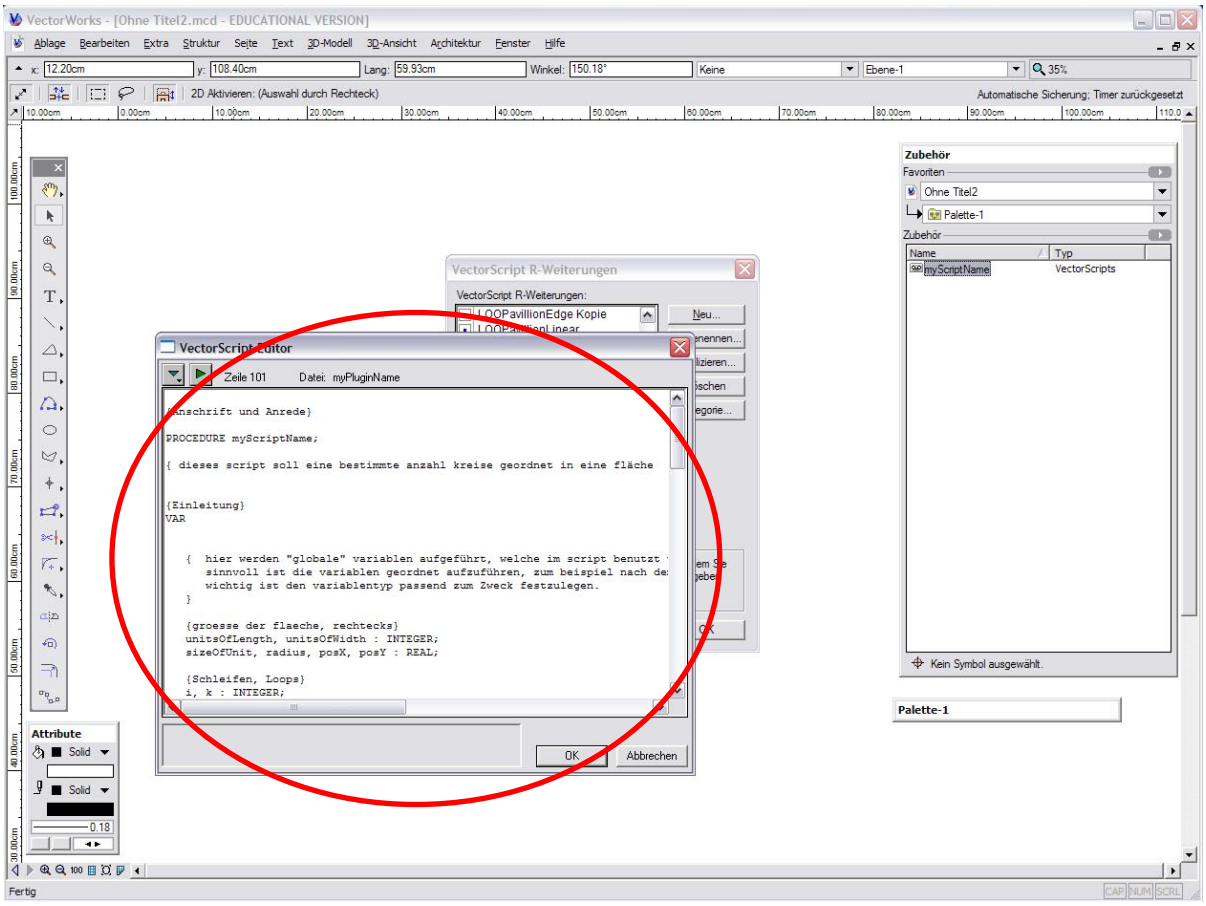
3. Plug In benennen und Typ festlegen..



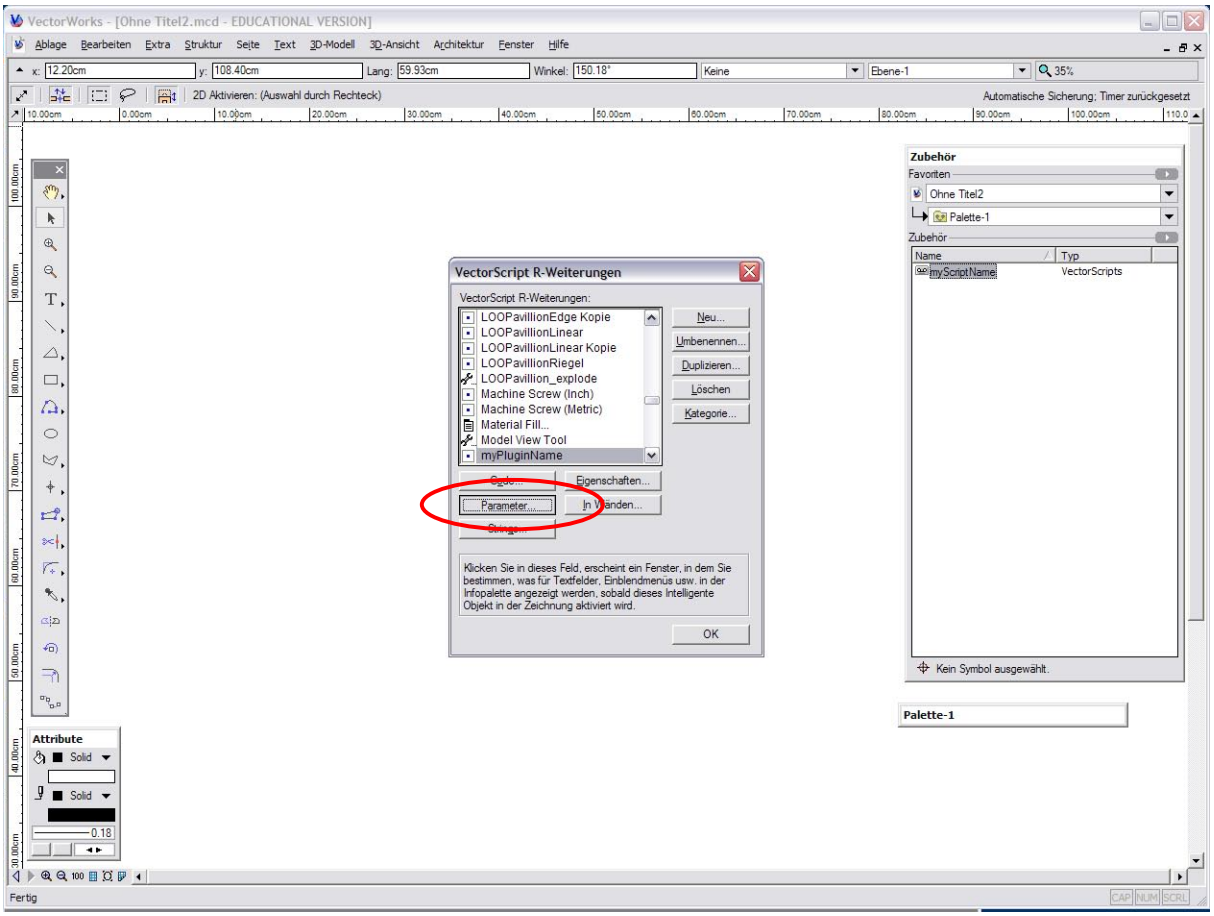
4. Plug In VectorScript Code



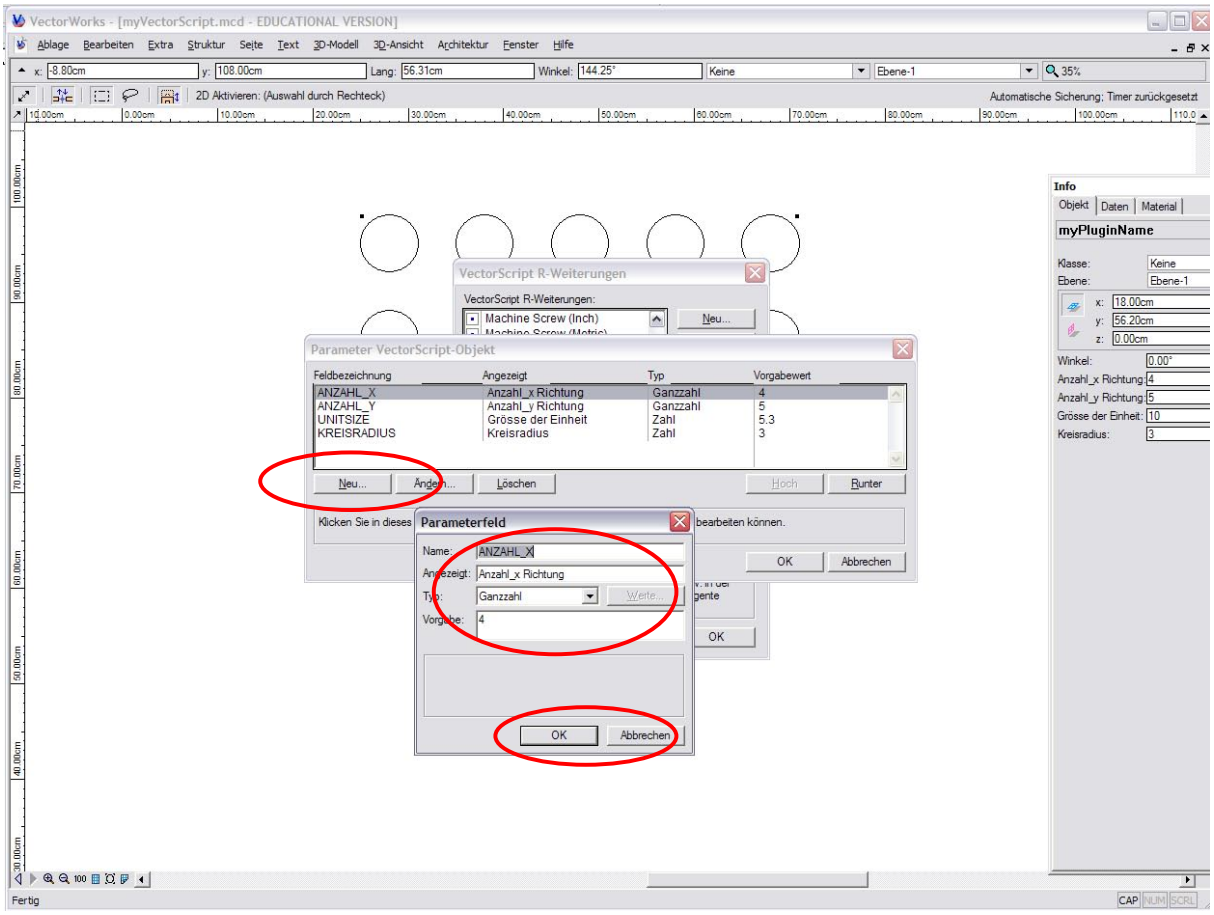
5. Das Script aus dem Zwischenspeicher wird nun eingefügt



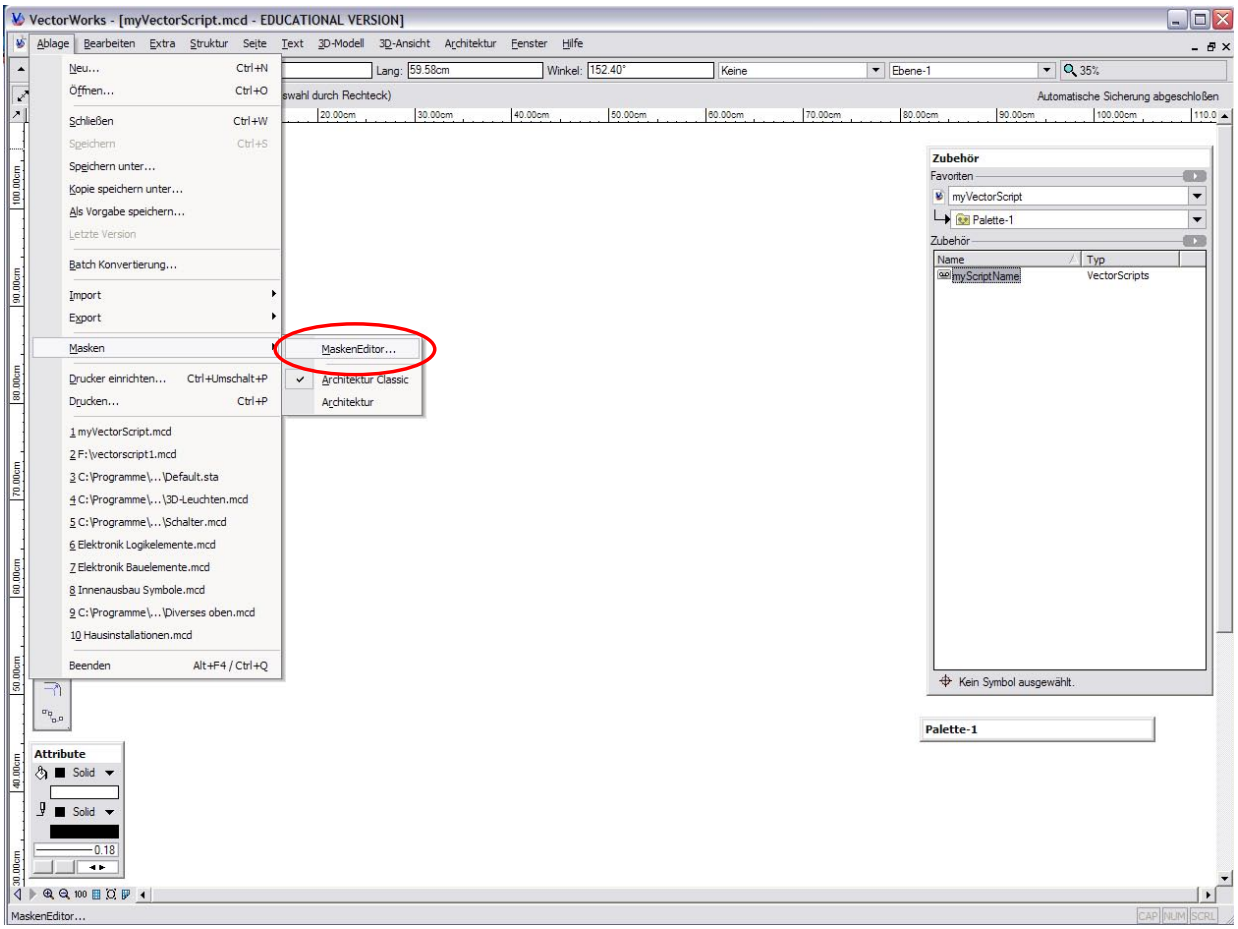
6. Die Plug In Parameter



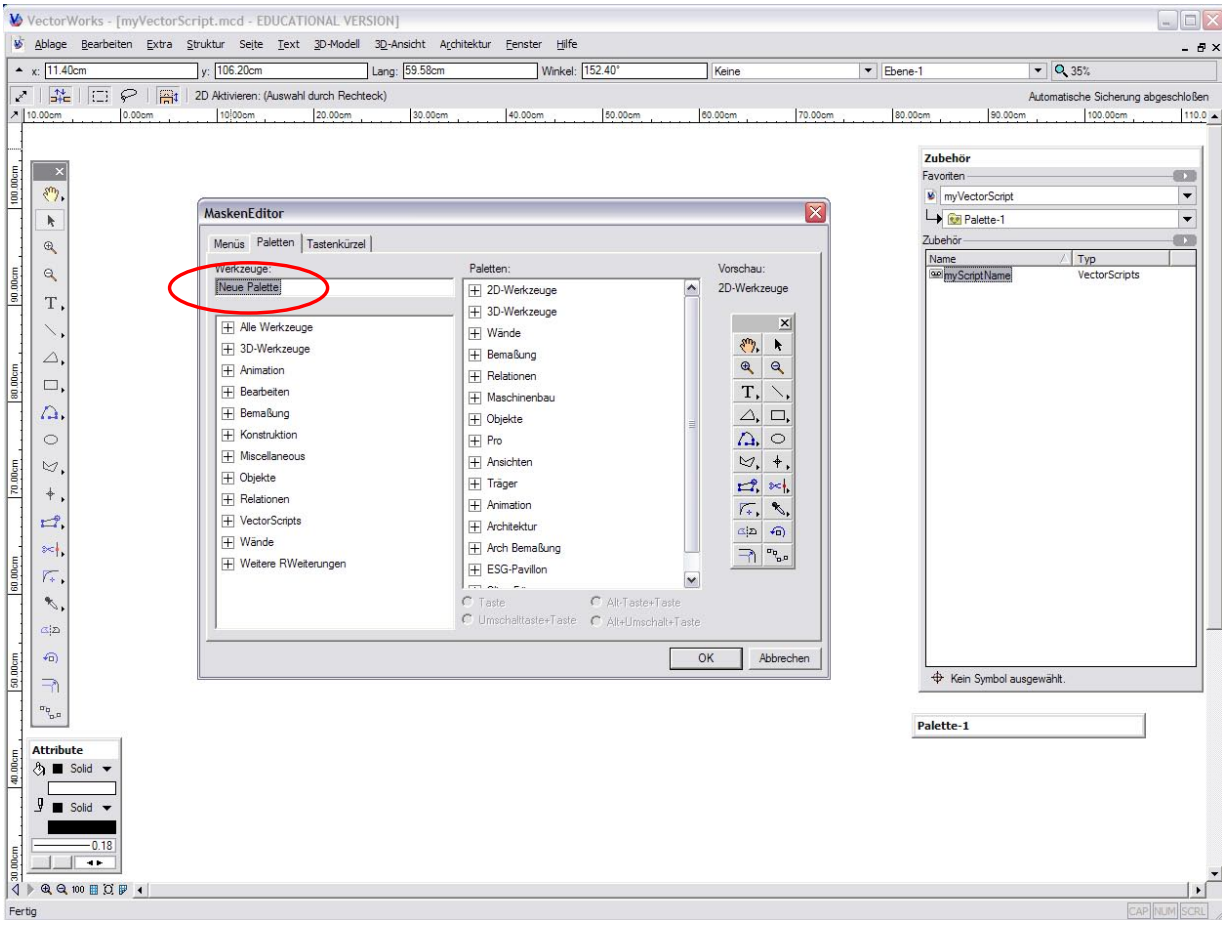
7. Plug In Parameter eingeben



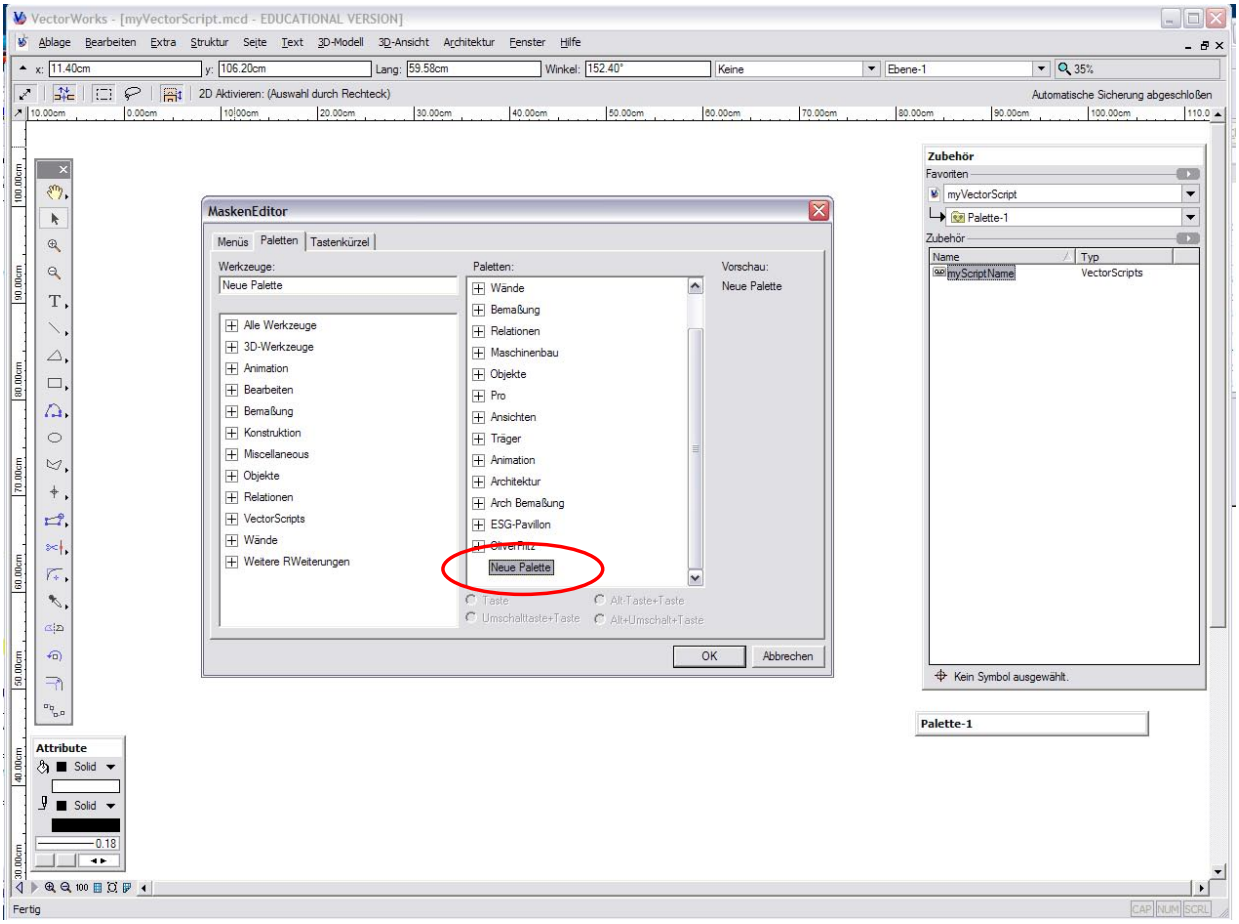
8. Plug In in den Worspace einbinden...



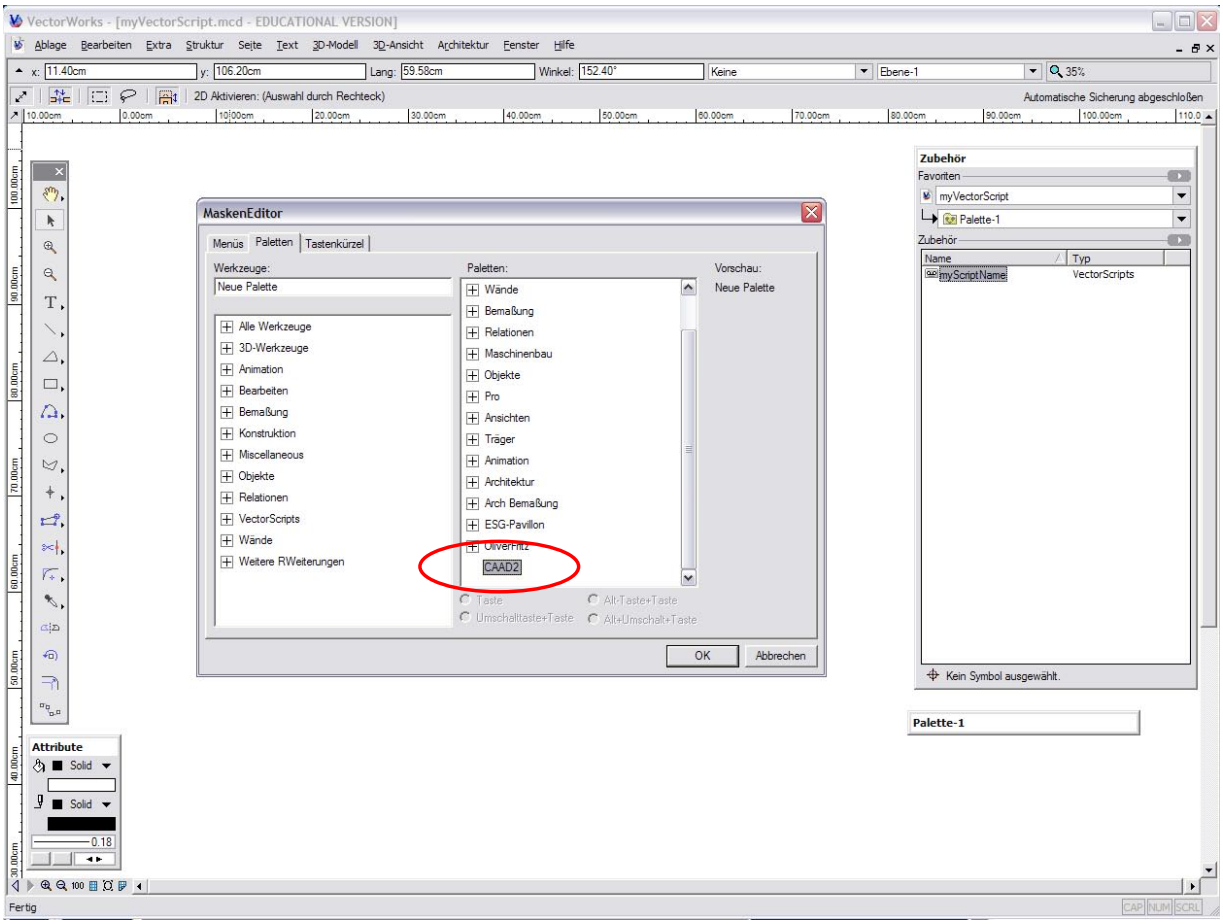
9. Neue Palette einrichten



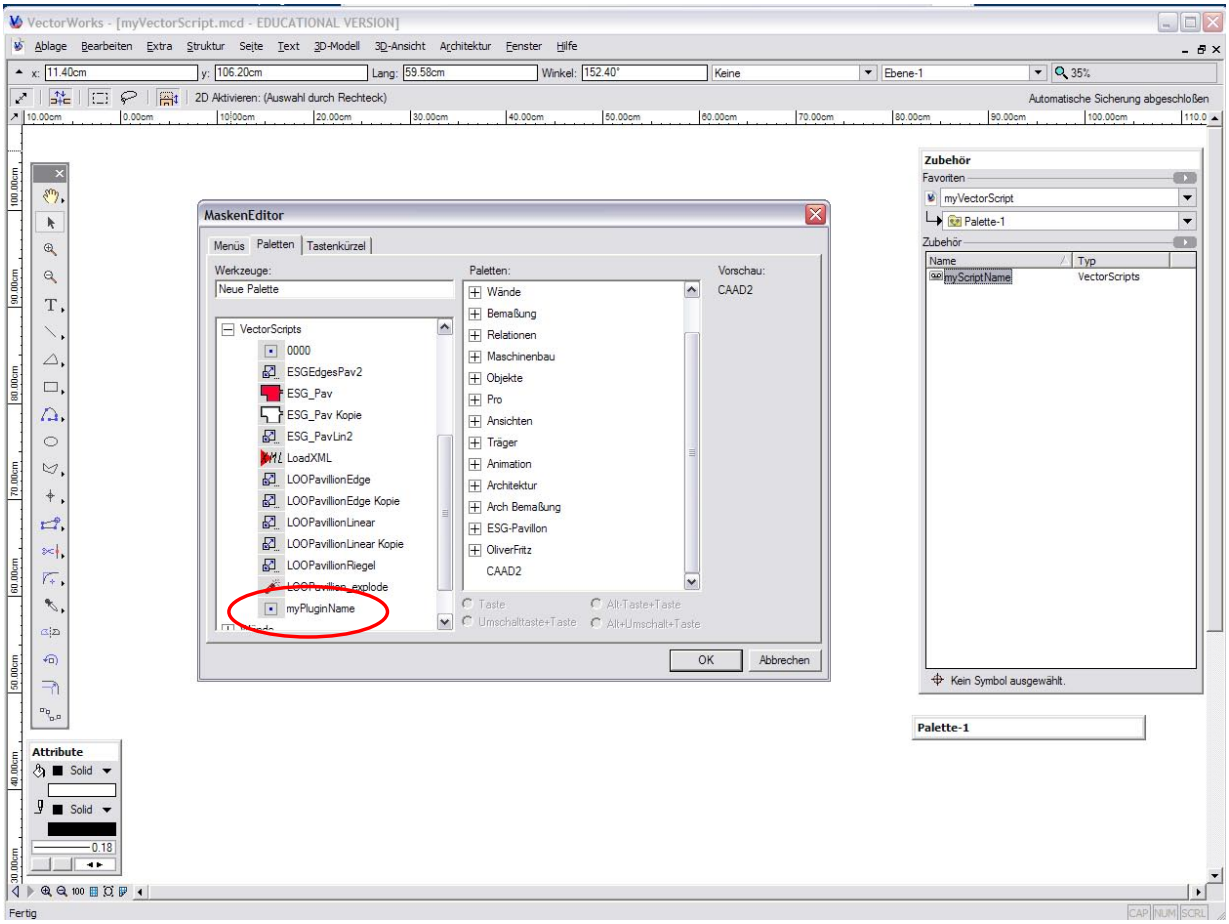
10. neue Palette benennen..



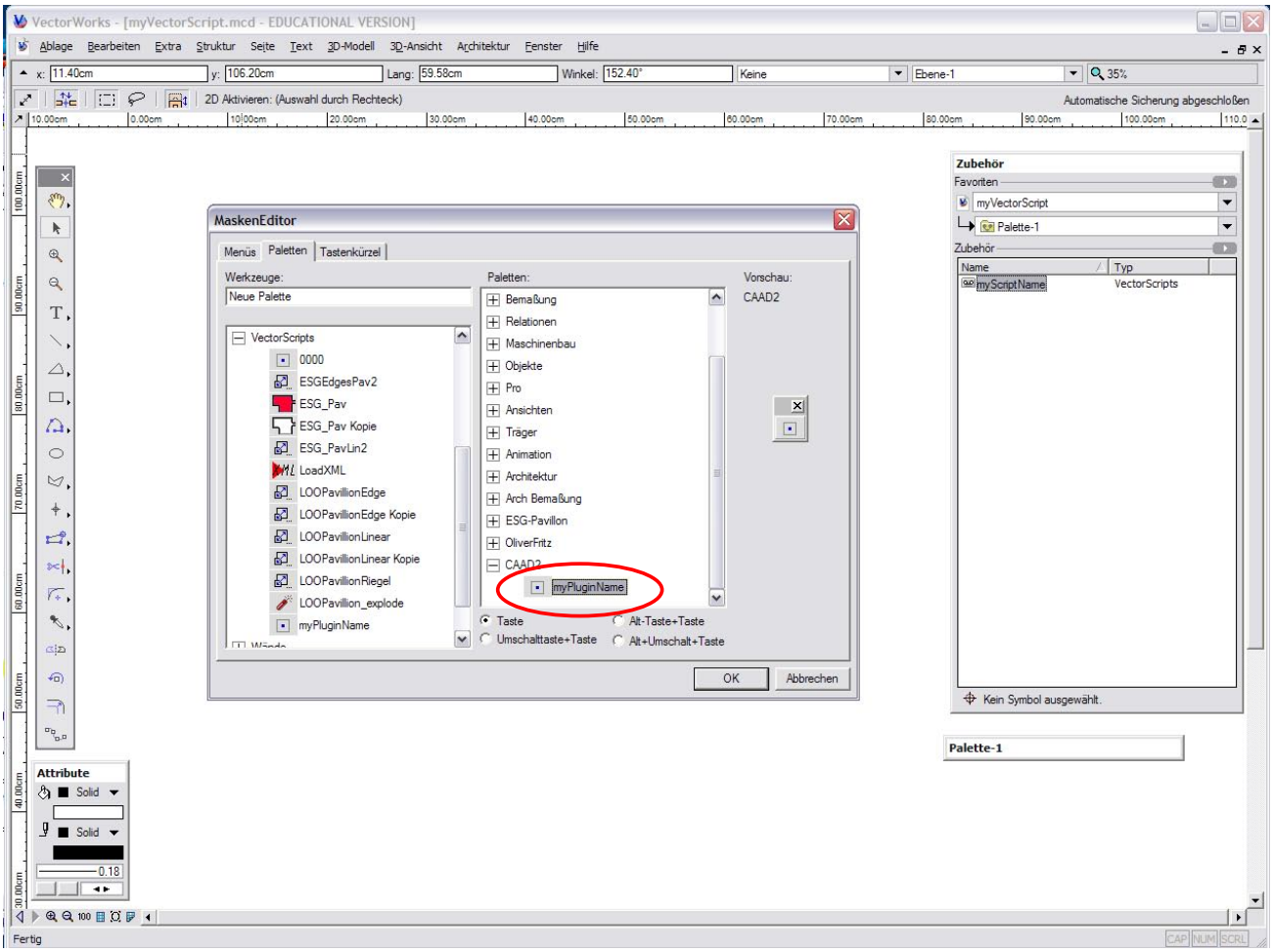
zum Beispiel CAAD2



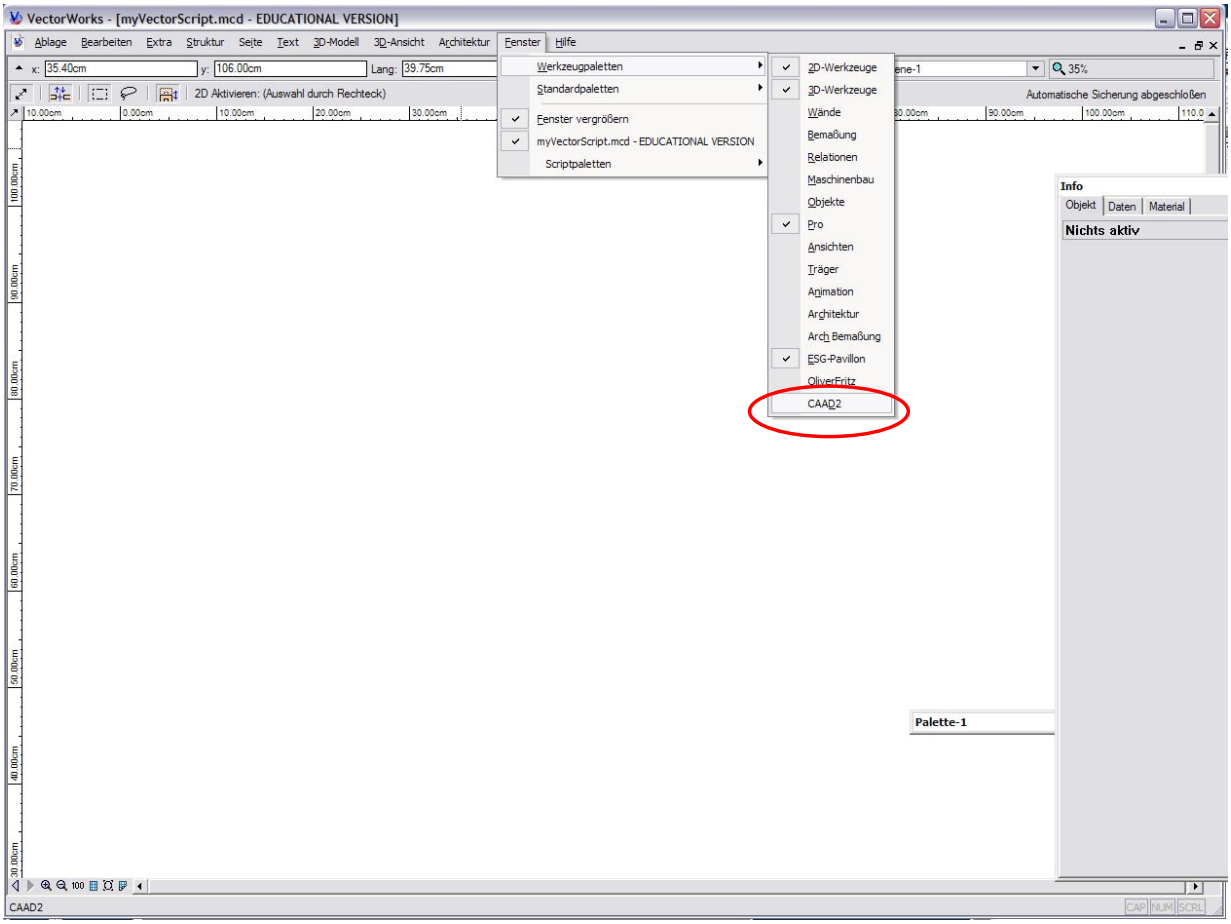
11. das Plug In suchen



...und in dei CAAD2 Palette ziehen



12. die neue CAAD2 Palette aufrufen...



..FERTIG !