

Bild 1.5: Schrift erstellen

Schrift erstellen

Im Menü *Tools* erscheint mir der Eintrag *Schrift erstellen* besonders interessant. Er führt zu einem Menü, in welchem Sie eine der gängigen PC-Schriftarten zuzüglich ihrer Font-Größe auswählen können. Beim Klick auf OK wird ein Verzeichnis im aktuellen Arbeitsverzeichnis abgelegt, das die Daten der gewählten Schrift enthält. Mit ein paar Programmzeilen, auf die wir an geeigneter Stelle noch zu sprechen kommen, kann diese Schrift dann während der Laufzeit des Programms auf den Bildschirm gebracht werden. Ein sehr nützliches Tool, da die von Processing während der Laufzeit erzeugte Schrift nach meinem ästhetischen Empfinden nicht besonders ansprechend aussieht, da sie irgendwie verschmiert erscheint.

Referenz

Im Hilfe-Menü finden Sie unter dem Eintrag *Referenz* eine Liste aller Anweisungen und Operatoren, die in einem Processing-Programm zur Anwendung kommen. Dazu ist jedoch eine Internetverbindung erforderlich. Die Gruppierung ist etwas gewöhnungsbedürftig, aber mit etwas Geduld findet man schließlich alles, was man sucht. Ein Klick auf den gesuchten Begriff führt zu weiteren Webseiten mit Erklärungen und Programmbeispielen, die jedoch nicht immer das zeigen, was man gerade wissen möchte und daher oft eine weitere Durchforstung des Internets erforderlich machen.



Bild 1.6: Ausschnitt aus der Processing-Referenzseite

Programm als .exe exportieren

Das Verwandeln eines Quellcodes in eine exe-Datei bietet viele Vorteile: Erstens: Der Anwender kann den Code nicht sehen und daher auch keine Tricks und Kniffe des Programmierers für seine eigenen Zwecke verwenden. Eine Exe-Datei kann zweitens durch Doppelklick aus dem Explorer oder vom Desktop aus gestartet werden, ohne dass man vorher umständlich einen Editor öffnen muss. Leider läuft eine Processing-Exe-Datei jedoch nicht schneller, als wenn der Quellcode von Editor aus gestartet wird. Dies ist jedoch nicht weiter schlimm, da die Geschwindigkeit eines vom Editor aus gestarteten Processing-Programms bereits, wie schon erwähnt, sehr hoch ist.

So wird's gemacht

Um einen Processing-Code in eine exe-Datei zu verwandeln, muss zunächst einmal eine wichtige Bedingung erfüllt sein: Auf Ihrem Computer muss das

Java Runtime Environment

installiert sein. Ist dies nicht der Fall, so informiert Sie Ihr Compiler spätestens beim Klicken auf den *Exportieren*-Button darüber. Das *Runtime Environment* können Sie über Ihren Webbrowser suchen lassen und dann auf die übliche Weise auf Ihrem Computer installieren. Das Ganze geschieht weitgehend automatisch, wenn Sie den Anweisungen auf dem Bildschirm folgen. Zum Zeitpunkt, als ich an diesem Manuskript geschrieben habe, handelte es sich um die Version 8. Diese Nummer kann sich im Laufe der Zeit ändern. Machen Sie nicht den Fehler, den ich gemacht habe: Beim Suchen bin ich auf eine Version mit höherer Nummer (10) gestoßen und dachte mir: 10 ist jedenfalls neuer als 8, also nehme ich die. Das war jedoch ein Irrtum: Es hat nicht funktioniert.

Wenn Sie also die richtige Version installiert haben, starten Sie den Editor mit dem zu exportierenden Programm, öffnen dort das *Datei*-Menü und klicken auf *Exportieren* (**Bilder 1.7 und 1.8**). Daraufhin öffnet sich das Menü mit der Bezeichnung *Export Optionen*. Dort klicken Sie auf das von Ihnen verwendete Betriebssystem und dann auf *Exportieren*. Im

Verzeichnis, in welchem sich Ihre pde-Datei befindet, erscheinen nun (auf meinem Rechner mit der 64-Bit-Windows 7-Version), zwei zusätzliche Verzeichnisse: *Application.windows32* und *Application.windows64*. In beiden Verzeichnissen befindet sich eine exe-Datei mit dem gleichen Namen wie die Datei des Quellcodes, mit dem Unterschied, dass sie die Erweiterung exe statt pde besitzen. In dem betreffenden Verzeichnis befinden sich noch weitere Verzeichnisse mit Ressourcen.

Hinweis 1: Wenn Sie eine exe-Datei weiter geben, muss stets das komplette, übergeordnete Verzeichnis weitergegeben werden.

Hinweis 2: Vergessen Sie nicht, eventuelle Dateien, die vom Programm aus ohne absolute Pfadangabe während der Laufzeit geladen werden, mit in das Verzeichnis zu geben. Ich spreche aus Erfahrung: Ich glaubte am Anfang, dass das Verwandeln in exe-Dateien auf meinem Computer nicht funktioniert, weil nach dem Start der exe-Datei stets ein Absturz erfolgte. Irgendwann war mir dann klar, dass das nach dem Programmstart automatisch zu ladende Hintergrundbild nicht gefunden wurde, was dann zu einem Absturz führte. Es wurde **nicht**, wie ich annahm, in die exe-Datei eingebettet.

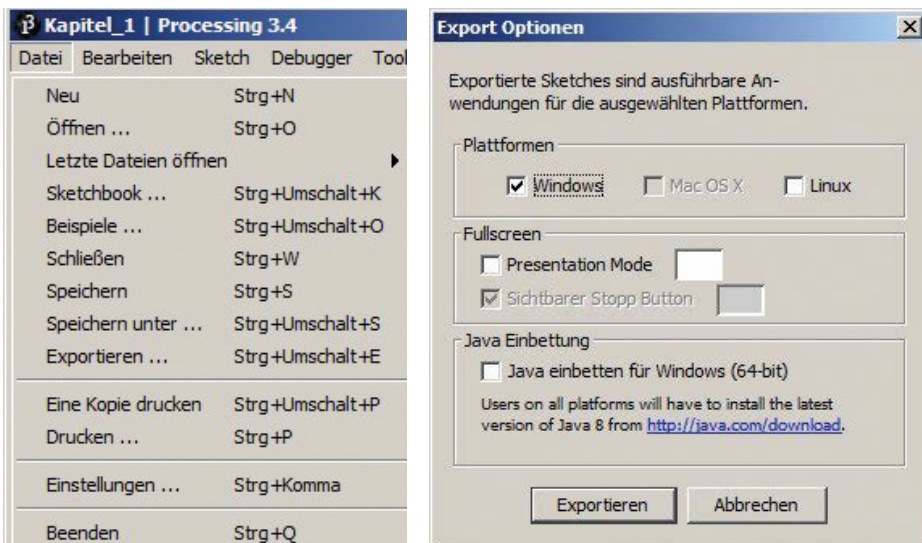


Bild 1.7 und 1.8: Erzeugen von Processing-exe-Dateien.

Bei längeren Listings ist die Suchfunktion sehr hilfreich. Wenn Sie im Menü *Bearbeiten* auf den Eintrag *Suchen* klicken, öffnet sich das in Bild 1.9 gezeigte Fenster *Suchen*. Geben Sie in der Zeile *Suche* den



Bild 1.9: Suchfunktion

gesuchten Begriff ein und klicken Sie auf den Button *Suchen*. Der erste im Listing gefundene Begriff wird markiert dargestellt. Nach jedem erneuten Klick erscheint der als nächstes gefundene Suchbegriff. Bei der Funktion *Suchen* und *Ersetzen* sollte man vorsichtig sein: Hier werden alle Begriffe im kompletten Listing durch den Begriff ersetzt, der in *Ersetzen* durch eingetragen wurde. Mir ist es leider nicht gelungen, eine Lösung zu finden, bei der nur die Begriffe in einem per Maus markierten Abschnitt ersetzt werden. Auch die im Eintrag *Suche in Auswahl* gefundene Lösung hat nicht das Gewünschte gebracht: Statt die in der per Maus markierten Auswahl gesuchten Begriffe zu ersetzen, wurde der komplette markierte Bereich gelöscht und durch das in *ersetzen durch* eingetragene Wort ersetzt. Das ist natürlich wenig hilfreich. Ich vermute, dass es sich hier ganz einfach um einen Fehler im Editor handelt.

Manchmal ist es jedoch notwendig, dass in einem längeren Code-Abschnitt ein Begriff durch einen anderen ersetzt werden soll – zum Beispiel, wenn eine zweite, ähnliche Funktion (oder ein Teil einer Funktion) erzeugt werden soll, in welcher jedoch ein anderer Variablen- oder Array-Name vorkommt. Was tun, wenn die Funktion *Suchen und Ersetzen* aber nicht richtig funktioniert? Die Lösung: Man kann den gesamten betreffenden Code mit *Copy & Paste* kopieren und dann den alten Variablennamen wiederum per *Copy & Paste* an den betreffenden Stellen durch den neuen Namen ersetzen. Dies kann bei einem umfangreicheren Code manchmal sehr lange dauern. In solchen Fällen kann das Textprogramm *Word* sehr hilfreich sein: Kopieren Sie den betreffenden Code-Abschnitt in eine leere *Word*-Seite. Stören Sie sich nicht daran, dass die Formatierung nun etwas unübersichtlich geworden ist. Makieren Sie den gesamten Text auf der *Word*-Seite und ändern Sie den Code durch die *Suche und Ersetzen* Funktion von *Word*. Nun können Sie den geänderten Code per *Copy & Paste* wieder zurück in den *Processing Editor* kopieren. Die Formatierung besitzt nur wieder das ursprüngliche, übersichtliche Aussehen.

Bei solchen *Copy-&Paste*-Aktionen sollte man jedoch sehr vorsichtig sein, dass man dabei nicht irgendwelche Codezeilen verdoppelt oder löscht. Der Editor reagiert in solchen Fällen sehr empfindlich und gibt Fehlermeldungen aus, die nicht wirklich weiter helfen.

Autoformatierung

Im Menü bearbeiten gibt es noch ein weiteres Feature, das mir interessant erscheint: Autoformatierung. Beim Programmieren hat man nicht immer Zeit, die Zeilen jedesmal auch so einzurücken, dass alles besonders ästhetisch und übersichtlich aussieht. Ein richtig eingerückter Code erleichtert jedoch die Übersicht. Dies ist auch wichtig, wenn man seinen eigenen Code zu einem späteren Zeitpunkt erneut bearbeiten möchte. Dann ist es oft schwierig, sich in frühere Gedankengänge wieder einzuarbeiten. Mit Autoformatierung wird ein unordentlich geschriebener Code sozusagen verschönert und übersichtlicher gestaltet, so dass er wie aus dem Lehrbuch kopiert aussieht. Was mir jedoch nicht gefällt, ist die Tatsache, dass dabei die öffnenden geschweiften Klammern an das Ende einer Zeile gesetzt werden. Auch wenn man dadurch eine Zeile gespart hat: Übersichtlich finde ich dies nicht – aber das ist wohl letztlich Geschmacksache. Unterm Strich auf jeden Fall ein nützliches Tool. In diesem Buch habe ich übrigens meine eigene, mir am übersichtlichsten erscheinende Schreibweise eingeführt: Jede geschweifte Klammer bekommt (in den meisten Beispielen) ihre eigene Zeile. Dies kostet zwar ein wenig mehr Platz, aber es lohnt sich, da der Code dann wesentlich übersichtlicher wird.

In den verschiedenen Menüs haben Sie sicher schon gesehen, dass es noch viele andere Features gibt. Wollen wir es an dieser Stelle bei der Besprechung der mir persönlich am interessantesten erscheinenden Menüpunkte belassen. Bei einem Klick auf *Optimieren* (Sketch- Menü) ist mir sogar der Editor abgestürzt. Also Vorsicht beim weiteren Erkunden und Experimentieren!