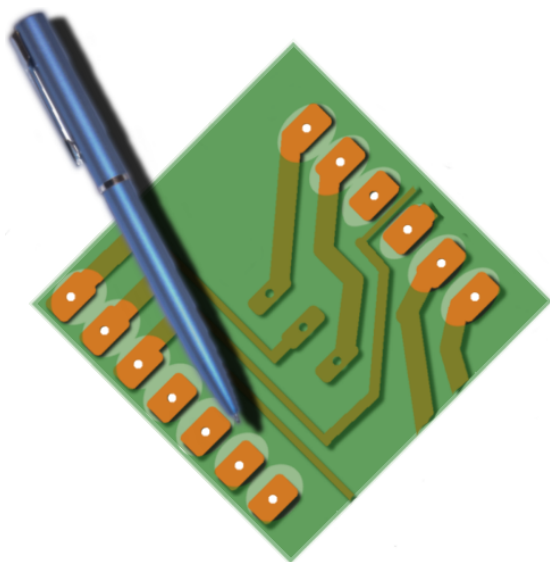


# FidoCadJ 0.23

## le manuel de l'utilisateur

Davide Bucci



23 décembre 2009



Cette oeuvre est sujette à la Creative Commons Public License version 3.0 ou plus récente. L'énoncé intégral de cette licence est disponible à l'adresse

[http ://creativecommons.org/licenses/by-nc-nd/3.0/deed.fr](http://creativecommons.org/licenses/by-nc-nd/3.0/deed.fr).

Vous êtes libres de reproduire, distribuer, communiquer ou exposer à un publique, représenter, exécuter et réciter cette oeuvre aux conditions suivantes :

**ATTRIBUTION** Vous devez citer le nom de l'auteur original de la manière indiquée par l'auteur de l'oeuvre ou le titulaire des droits qui vous confère cette autorisation (mais pas d'une manière qui suggérerait qu'ils vous soutiennent ou approuvent votre utilisation de l'oeuvre).

**NON COMMERCIAL** Vous n'avez pas le droit d'utiliser cette création à des fins commerciales.

**PAS D'OEUVRES DÉRIVÉES** Vous n'avez pas le droit de modifier, de transformer ou d'adapter cette création. Chaque fois que l'on utilise ou on distribue cette oeuvre, il faut le faire selon les termes de cette licence, qui doit être communiquée avec clarté.

Chacune de ces conditions peut être levée si vous obtenez l'autorisation du titulaire des droits sur cette oeuvre (Davide Bucci).

Les noms commerciaux, les logos, les marques déposées appartiennent aux propriétaires respectifs.

## RÉSUMÉ

---

Ce document est le manuel officiel de FidoCadJ. Après une brève présentation de l'histoire de ce logiciel et de sa philosophie, nous allons voir les notions principales nécessaires pour dessiner un simple schéma électrique et un circuit imprimé. Ce manuel s'achève avec des détails techniques peu documentés jusqu'à présent, comme la description complète du format de fichier utilisé par FidoCad et par conséquence par FidoCadJ.

## REMERCIEMENTS

---

Beaucoup de personnes ont utilisé le logiciel (ou une de ses versions préliminaires) et il m'ont fait parvenir leur impressions. Je veux donc remercier ici pour leur conseils les assidus du groupe de discussion `it.hobby.elettronica`. Ce logiciel a été testé sous Linux grâce la patience de Stefano Martini, qui a été un fin chercheur de bogues. En outre, je veux remercier Olaf Marzocchi et Emanuele Baggetta pour leur tests sous MacOSX.

Je remercie F. Bertolazzi pour m'avoir motivé à faire en sorte que le logiciel soit utilisable et qui a assemblé avec patience la librairie CadSoft Eagle permettant l'exportation vers ce logiciel. Je remercie Celsius, qui a testé le fonctionnement du logiciel pour la réalisation de circuits imprimés et les librairies, ainsi qu'Andrea D'Amore, pour ses conseils sur comment améliorer l'apparence du logiciel sous Apple Macintosh, à partir de la version 0.21.1. Un merci à Roby IZ1CYN, qui a écrit le paragraphe [A.2.2](#) de ce manuel. Je remercie aussi Pasu, qui a traduit tout ce manuel en anglais, pour favoriser la diffusion de ce logiciel. Il m'a aussi aidé à trouver plusieurs fautes dans la version anglaise de l'interface.

## NOTE POUR LA VERSION FRANÇAISE DU LOGICIEL

---

FidoCadJ est un logiciel qui est né au sein d'une communauté italienne d'utilisateurs. Cependant, j'habite en France depuis un certain nombre d'années et j'ai choisi de faire un effort pour traduire le logiciel en français. Comme mon temps est limité, vous pouvez constater que pas tout a été traduit complètement. Par exemple, les figures de ce manuel se réfèrent en grande partie à la version italienne. Plus urgent, la librairie standard utilise plusieurs fois des descriptions en italien pour les symboles qu'il contient.

Je lance donc un appel à la communauté francophone : si vous repérez une erreur de traduction dans ce manuel, ou vous voulez participer à cette aventure, n'hésitez pas à me contacter. Je serai très heureux de recevoir de l'aide, afin que FidoCadJ commence à se diffuser aussi en France.

## LICENCE FIDOCADJ

---

Copyright © 2007-2009 Davide Bucci [davbucci@tiscali.it](mailto:davbucci@tiscali.it)

This program is free software : you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3 of the License.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY ; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see [http ://www.gnu.org/licenses/](http://www.gnu.org/licenses/).

## TABLE DES MATIÈRES

1	INTRODUCTION	1
1.1	La philosophie de FidoCadJ	1
1.2	Histoire du logiciel	1
2	LE DESSIN AVEC FIDOCADJ	4
2.1	Les instruments de dessin	4
2.2	Dessignons un simple schéma électrique	7
2.3	Les couches (layer)	11
2.4	La grille	12
2.5	Dessiner un simple circuit imprimé	12
2.6	Frecce e stili di tratto	17
2.7	Exportation	18
2.8	Options de la ligne de commande	19
2.9	Gestion des librairies	21
3	LE FORMAT DES DESSINS, DES MACROS ET DES LIBRAIRIES FIDOCAD	22
3.1	Description de l'en-tête	22
3.2	Le système de coordonnées	22
3.3	Éléments de dessin	22
3.4	Extensions de FidoCadJ	27
3.5	Tolérance aux erreurs de syntaxe	27
3.6	Le format des librairies	28
3.7	Librairies standard	29
4	CONCLUSION	30
A	EXTENSIONS SPÉCIFIQUES AUX SYSTÈMES D'EXPLOITATION	31
A.1	MacOSX	31
A.1.1	Extensions	31
A.1.2	Comment télécharger et exécuter FidoCadJ sous MacOSX	31
A.2	Linux	32
A.2.1	Estensions	32
A.2.2	Comment télécharger et executer FidoCadJ sous un système Linux	32
A.2.3	Sur n'importe quel système, depuis l'invite des commandes	32
A.2.4	Sur un système graphique	33
A.3	Windows	34
	INDEX	34



## TABLE DES FIGURES


FIG. 1	Une session de travail typique de FidoCadJ sous MacOSX Tiger. Le logiciel est localisé en langue française. Dans l'appendice A, nous allons décrire les particularités de la version spécifique pour Macintosh. 5
FIG. 2	Comment se présente FidoCadJ avec le «look and feel» Metal. 5
FIG. 3	La possibilité de recherche à l'intérieur des librairies chargées. 5
FIG. 4	La fenêtre des paramètres du texte en FidoCadJ 7
FIG. 5	Voici notre objectif : un miroir de courant réalisé avec des transistors NPN. 7
FIG. 6	Commençons à positionner les deux transistors. 8
FIG. 7	Nous pouvons sélectionner le transistor sur la gauche et le retourner avec  . 8
FIG. 8	Nous sommes trop en haut : nous pouvons tous sélectionner et déplacer vers le bas de la fenêtre. 9
FIG. 9	Le circuit, presque terminé. 10
FIG. 10	Voici le circuit terminé, dans toute sa splendeur. 10
FIG. 11	Un très simple étage amplificateur à émetteur commun. 13
FIG. 12	Les composants sont placés sur la plaquette. 13
FIG. 13	Nous avons rajouté les régions de masse et d'alimentation positive à l'aide de deux polygones. 14
FIG. 14	Nous avons rajouté les connexions de masse et l'alimentation positive, à l'aide de deux polygones. 14
FIG. 15	Voici le circuit imprimé, presque terminé. 15
FIG. 16	Le travail terminé, avec toutes les sérigraphies. 15
FIG. 17	Le circuit imprimé, tel qu'il apparaît imprimé en miroir sur une page en format ISO-UNI A4. 16
FIG. 18	Un schéma électrique d'un GIC dans lequel on a utilisé quelques extensions de FidoCadJ. 17
FIG. 19	La fenêtre des paramètres d'un tracé de Bézier, dans un dessin FidoCadJ. 18
FIG. 20	L'aspect de FidoCadJ sous MacOSX, en utilisant le «look and feel» Motif. 21
FIG. 21	La figure 18 comme apparaîtrait si montrée avec FidoCad. 28
FIG. 22	La fenêtre de gestion des permissions, dans la distribution Ubuntu 8.04. 33

FIG. 23            Sélectionner l'exécution de la machine virtuelle  
Java, dans la distribution Ubuntu 8.04.    33

## LISTE DES TABLEAUX

---

TAB. 1	Résumé des commandes de dessin disponibles avec FidoCadJ. La touche montrée dans la colonne la plus à gauche permet de sélectionner rapidement depuis le clavier l'opération désirée. Un click avec la touche droite dans la modalité d'insertion des éléments permet d'afficher la fenêtre de modification des propriétés.    6
TAB. 2	Liste des formats d'exportation disponibles avec FidoCadJ.    20
TAB. 3	Présence des flèches aux extrêmes d'un segment ou d'un tracé de Bézier, selon la valeur du code a.    23
TAB. 4	Style des flèches, selon la valeur du code b.    23
TAB. 5	Fonctions des bits dans le style du texte.    24

## INTRODUCTION

---

Dans ce chapitre, nous allons présenter rapidement FidoCadJ. En particulier, nous allons ré-parcourir la philosophie qui est à la base de ce logiciel, ainsi que l'histoire de sa création et son développement.

### 1.1 LA PHILOSOPHIE DE FIDOCADJ

FidoCad (sans la J à la fin du mot) est un logiciel de dessin vectoriel, particulièrement adapté à être utilisé pour tracer des schémas électriques et des circuits imprimés. Il s'est diffusé particulièrement dans la communauté Usenet italienne, à partir de la fin des années 1990. Il peut être téléchargé gratuitement (en version pour Windows) depuis la page de son auteur, Lorenzo Lutti :

<http://www.enetsystems.com/lorenzo/fidocad.asp>

Le fichier généré par ce logiciel contient seulement des codes en format texte et est très compacte. Ce détail le rend très adapté à être utilisé comme format d'échange dans les messages des groupes de discussion quand des dessins sont nécessaires. Malheureusement, FidoCad existe seulement dans la version pour Windows. Qui utilise Linux peut utiliser WInE pour sortir de l'impasse, mais qui utilise un Macintosh doit trouver une solution alternative. J'ai voulu donner une petite contribution, en écrivant FidoCadJ (avec la J à la fin, cette fois). Cet éditeur est écrit en Java, est complètement multi-plateforme et permet de visualiser et de modifier les dessins en format FidoCad.

Qui a déjà utilisé FidoCad dans le passé devrait se trouver rapidement à son aise avec FidoCadJ, car la majorité des commandes sont très similaires. Naturellement, les fonctionnalités de FidoCadJ sont moins développées que celles de FidoCad, en étant ce dernier développé depuis plus longtemps. En particulier, la gestion des libraires et des macros est très essentielle. Mon but a été de fournir une solution minimaliste aux exigences de dessin et de projet de petits circuits. Le tout en respectant la philosophie du logiciel originel, qui est de fournir un outil simple et essentiel.

Bien évidemment, mon but a été d'arriver à une compatibilité totale à niveau des fichiers générés par FidoCadJ et FidoCad pour Windows.

Parmi mes intérêts, on trouve la typographie à l'ordinateur et en particulier le logiciel  $\text{\LaTeX}$ . Pour cette raison, FidoCadJ permet (et ceci à différence de FidoCad) de réaliser l'exportation d'un dessin en plusieurs formats vectoriel et entre autre en Postscript encapsulé (connu aussi comme EPS). Une autre possibilité intéressante qui est mise à disposition à partir de la version 0.21 de FidoCadJ est l'exportation de schémas électriques vers des scripts exécutables par CadSoft Eagle. À partir de la version 0.22, FidoCadJ permet d'exporter vers le format PDF.

### 1.2 HISTOIRE DU LOGICIEL

Je suis passionné d'électronique depuis beaucoup de temps. Quand j'ai commence à fréquenter des groupes de discussion Usenet italiens, je me suis aperçu que la majorité des schémas électriques était fournis

*De toute façon, je pense qu'il vaut mieux se mettre à l'oeuvre, plutôt que se plaindre continûment que sous un système d'exploitation alternatif à Windows il n'y a pas tel ou tel autre programme.*

*En réalité, il était depuis 1993 que j'avais l'idée d'écrire un petit logiciel de dessin vectoriel 2D.*

en utilisant le format du logiciel FidoCad pour Windows, au lieu que faire des graphismes ASCII. N'utilisant plus Windows depuis plusieurs années, j'ai voulu faire un effort pour combler cette lacune.

La première chose que j'ai faite a été d'étudier en détail le format utilisé par FidoCad et écrire une applique Java, appelée FidoReadJ, capable d'interpréter et de montrer un circuit à l'intérieur d'une page web. J'ai donc fait pas mal de «reverse engineering» et j'ai cherché un peu partout pour me rendre compte des possibilités et des limitations du format. J'ai aussi téléchargé et étudié les sources en C++ (très lisibles et propres) de FidoCad, mises à disposition par Lorenzo Lutti. Tout ceci avait lieu plus ou moins vers mars 2007. Quelques mois plus tard, l'appliquette était en ligne sur mon site et était en train d'être utilisée et testée à fond par une partie de la communauté qui fait partie des groupes it.hobby.electronica et it.hobby.fai-da-te.<sup>1</sup>

En disposant désormais d'un interprète du format, le travail qui restait était de mettre à point l'interface utilisateur. La plus grande partie du travail a été réalisé en plusieurs étapes, entre janvier et juillet 2008. FidoCadJ n'est donc pas une adaptation ou un «porting» de FidoCad pour Windows, mais plutôt une ré-écriture totale du logiciel. À partir de la version 0.21, apparue en janvier 2009, FidoCadJ met à disposition des extensions au format FidoCad originel. Beaucoup d'attention a été faite pour que une compatibilité à l'arrière soit maintenue.

Mon choix d'utiliser Java a été motivé par le fait que dans ces dernières années j'ai changé plusieurs systèmes d'exploitation et d'architectures différentes. Je ne voulais donc pas me lier de façon indissoluble à un système en particulier. Pour cette raison, l'effort d'apprendre le framework Cocoa aurait probablement été contrebalancé par un aspect bien meilleur du logiciel sous MacOSX, mais il aurait rendu FidoCadJ non portable. Finalement, je ne suis pas du tout un informaticien et le temps que je passe à apprendre le langage à la mode à un instant donnée je ne le passe pas à faire de l'électronique.

Pour le reste, une simple analyse de la structure du code source de FidoCadJ révèle que je ne suis pas du tout un puriste de Java et de la programmation à objets. Beaucoup de solutions sont plus pragmatiques que élégantes.

Ce qui compte, plus que le choix d'un langage ou un autre, est l'impression d'utilisation que l'utilisateur a du produit fini. Pour cette raison, je suis toujours à l'écoute de vos suggestions, pour identifier les directions où il faut améliorer FidoCadJ. Pour résumer, je ne crois pas que Java soit la panacée de tous les maux et non plus un langage parfait. Je crois par contre qu'il est dénigré par beaucoup de personnes parce qu'il y a beaucoup de logiciels de mauvaise qualité qui ont été écrits en ce langage. Sans avoir des prétention de perfection et en restant dans le contexte de mes possibilités informatiques, je veux faire en sorte que FidoCadJ NE soit PAS un logiciel de mauvaise qualité. Encore une fois, je suis à votre écoute !

Depuis novembre 2009, j'ai ouvert un projet FidoCadJ sur SourceForge. Depuis cette page, on peut télécharger les exécutables, ainsi que les manuels de FidoCadJ. Vous pourrez aussi participer activement au développement du logiciel, en accédant au code source, éventuellement avec Subversion, ou le navigateur SVN offert par SourceForge :

<http://fidocadj.svn.sourceforge.net/viewvc/fidocadj/>

<sup>1</sup> FidoReadJ est toujours à disposition, bien que pas complètement à jour, à l'adresse : <http://davbucci.chez-alice.fr/index.php?argument=electronica/fidoread/fidoreadj.inc>.

Pour participer au développement de FidoCadJ, pas besoin d'être des programmeurs experts : si vous le souhaitez, vous pourrez par exemple traduire le logiciel dans une langue différente de celles disponibles actuellement. Vous pourrez aussi lire avec attention les manuels et corriger les fautes de langue, des inconsistance ou erreurs. Vous pourrez aussi analyser la cohérence et préparer une traduction de la librairie standard. Vous pouvez aussi participer aux forums, écrire un commentaire sur le logiciel, suggérer des améliorations ou des bogues à corriger.

# 2

## LE DESSIN AVEC FIDOCADJ

*Les utilisateurs Mac  
invétérés  
remarqueront les  
menus sont à leur  
place.*

L'utilisation du logiciel devrait être plutôt intuitive pour qui a déjà utilisé un outil de dessin vectoriel. L'aspect du logiciel sous MacOSX est visible dans la figure 1;. Sous d'autres systèmes il y a quelques détails qui changent, mais la philosophie reste strictement la même. La figure 2 montre par exemple le résultat que l'on obtient en utilisant le «look and feel» Metal, préconisé par Sun. Nous allons voir quels sont les fonctionnalités et les éléments de dessin offerts. Ce derniers sont les fonctions géométriques de base, à partir desquelles chaque dessin FidoCadJ peut être obtenu.

### 2.1 LES INSTRUMENTS DE DESSIN

*Un peu ce qui se  
faisait dans les radios  
anciennes et les  
commutateurs des  
années 1970.*

Dans la barre des commandes (en haut dans la fenêtre du logiciel), on peut trouver des boutons qui permettent de créer et modifier un dessin. Le tableau 1 montre un résumé rapide des commandes disponibles et décrit les actions qui sont possibles pour chaque commande. Chaque bouton de la barre des commandes reste enfoncé, une fois que l'on a fait click dessus.

Il est donc possible entre autre de sélectionner quel élément de dessin nous allons introduire.<sup>1</sup> En haut à droite, une liste actionnable avec un click montre la couche de dessin activée, ainsi que les autres disponibles (la description de la fonction des couches se trouve au paragraphe 2.3).

L'aspect de la barre des commandes est partiellement personnalisable. Cet élément se trouve juste en dessous de la barre du titre des fenêtres, comme l'on peut voir dans les figures 1 et 2. En particulier, l'utilisateur peut choisir deux tailles d'icônes, et aussi peut cacher le texte. Cette dernière possibilité permet d'économiser la place sur l'écran pour la fenêtre du logiciel. Ces aspects de configuration du logiciel sont disponibles dans la fenêtre qui s'active dans le menu «Vue/Préférences»<sup>2</sup>. Les modifications seront activées en redémarrant FidoCadJ, car à priori il ne s'agit pas de quelque chose qui doit être changée tout le temps.

Sur la droite de la fenêtre principale, un arbre regroupe les éléments (appelés «macros»), mis à disposition par les librairies chargées dans le logiciel. Il suffit de sélectionner un élément de la librairie et ensuite faire click dans la zone de dessin pour l'insérer. Les librairies de FidoCad contiennent à leur intérieur tous les symboles électriques classiques, ainsi qu'une bonne collection d'empreintes pour les circuits imprimés. À partir de la version 0.22, un champ de texte avec une loupe est visible au dessus de la liste à arbre (voir en figure 3). Taper du texte à son intérieur permet d'effectuer rapidement une recherche d'un élément parmi ceux disponibles dans les librairies chargées. En utilisant ensuite les touches flèche haut/bas, on pourra naviguer parmi les éléments trouvés.

La figure 4 montre un exemple de ce que l'on obtient en faisant double click, en modalité sélection, sur un élément du dessin ; dans ce

<sup>1</sup> Pour avoir plus d'informations sur le code associé à chaque élément pendant sa mémorisation, vous pouvez consulter le paragraphe 3.3.

<sup>2</sup> Sauf sous MacOSX, où cette entrée se trouve dans le menu «FidoCadJ».

## 2.1. Les instruments de dessin

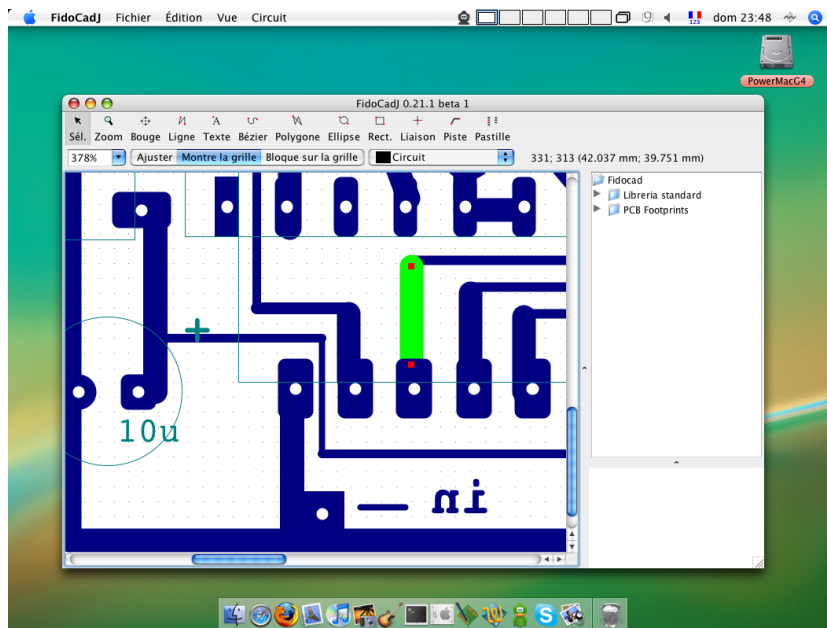


FIG. 1: Une session de travail typique de FidoCadJ sous MacOSX Tiger. Le logiciel est localisé en langue française. Dans l'appendice A, nous allons décrire les particularités de la version spécifique pour Macintosh.

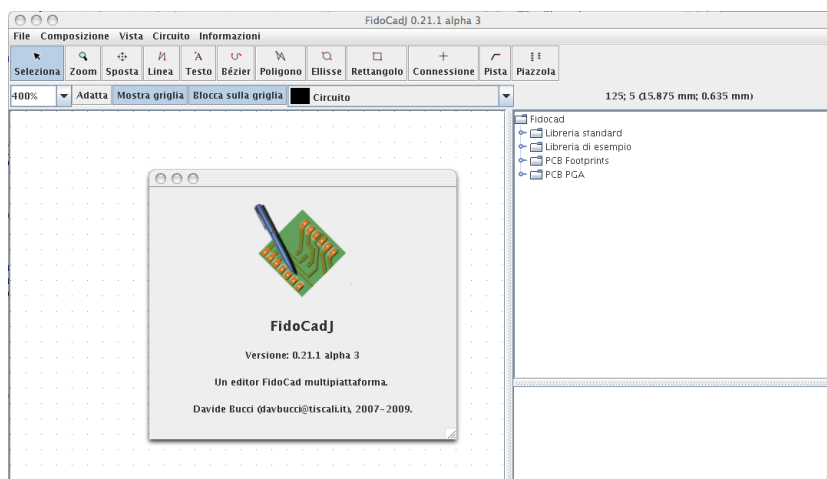


FIG. 2: Comment se présente FidoCadJ avec le «look and feel» Metal.

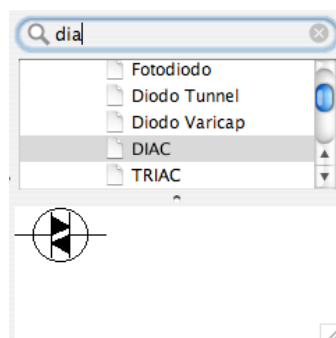














FIG. 3: La possibilité de recherche à l'intérieur des librairies chargées.

Touche	Commande	Utilisation
<span>A</span> ou <span>Espace</span>	 SELECTION	Sélection d'un ou plusieurs éléments graphiques. Appuyer sur <span>Control</span> (ou <span>Command</span> sous MacOSX) pour faire des sélections multiples, ou désélectionner un élément. Faire click et glisser la souris en gardant appuyée la touche, pour sélectionner plusieurs éléments dans une surface rectangulaire. Appuyer sur <span>R</span> pour faire pivoter les éléments sélectionnés. Appuyer sur <span>S</span> pour appliquer un effet miroir. Un double click sur un élément en montre les caractéristiques.
	 ZOOM	Faire click avec la touche gauche de la souris pour augmenter l'agrandissement. Faire click avec la touche droite pour le réduire.
	 BOUGE	Faire click et déplacer la souris pour faire bouger la partie du dessin montrée.
<span>L</span>	 LIGNE	Introduire une ligne, ou une série de lignes. Appuyer sur <span>Esc</span> , ou faire double click pour en terminer l'introduction.
<span>T</span>	 TEXTE	Introduire une ligne de texte.
<span>B</span>	 BÉZIER	Dessiner une courbe de Bézier.
<span>P</span>	 POLYGONE	Dessiner un polygone plein ou vide. Faire double click, ou appuyer sur <span>Esc</span> , per terminer l'introduction des points.
<span>E</span>	 ELLIPSE	Dessiner une ellipse pleine ou vide (maintenir enfoncé <span>Control</span> pour obtenir un cercle).
<span>G</span>	 RECTANGLE	Dessiner un rectangle plein, ou vide.
<span>C</span>	 CONNEXION	Dessiner une connexion électrique dans le schéma.
<span>I</span>	 PISTE C. I.	Dessiner une piste de circuit imprimé. La largeur par défaut de la piste peut être modifiée dans la fenêtre «Vue/Options dessin du circuit».
<span>Z</span>	 PASTILLE C. I.	Dessiner une pastille pour circuit imprimé. Les dimensions par défaut peuvent être modifiées dans la fenêtre «Vue/Options dessin du circuit».

TAB. 1: Résumé des commandes de dessin disponibles avec FidoCadJ. La touche montrée dans la colonne la plus à gauche permet de sélectionner rapidement depuis le clavier l'opération désirée. Un click avec la touche droite dans la modalité d'insertion des éléments permet d'afficher la fenêtre de modification des propriétés.



## 2.2. Dessinons un simple schéma électrique

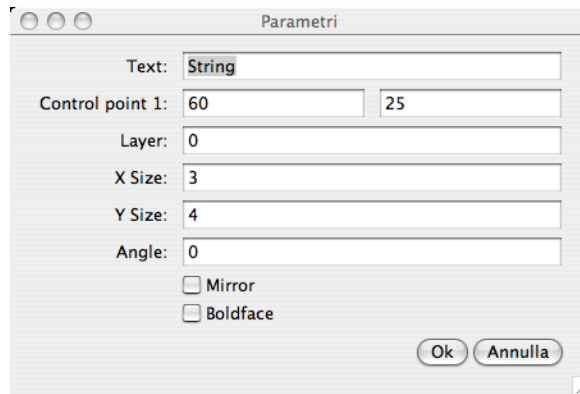


FIG. 4: La fenêtre des paramètres du texte en FidoCadJ

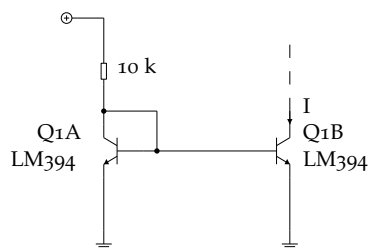


FIG. 5: Voici notre objectif : un miroir de courant réalisé avec des transistors NPN.

cas une ligne de texte. L'aspect de cette fenêtre ne va pas être strictement le même, selon quel élément graphique est en train d'être modifié.

## 2.2 DESSINONS UN SIMPLE SCHÉMA ÉLECTRIQUE

Pour comprendre comment le logiciel peut être utilisé, nous allons dessiner un simple schéma électrique, tel que celui montré en figure 5. Pour commencer, nous pouvons démarrer FidoCadJ, ou créer un nouveau dessin avec l'option «Fichier/Nouveau dessin».

Nous allons commencer donc par introduire dans le dessin les symboles des deux transistors, qui représentent un peu le cœur de notre schéma. Pour effectuer ceci, nous pouvons utiliser les macros mises à disposition par la librairie standard, qui est chargée automatiquement et dont les éléments sont affichés sur la droite de la fenêtre. La macro qui nous intéresse est appelé «NPN sans écran» et se trouve dans la catégorie «Diodes et transistors», à l'intérieur de la «Librairie standard». En faisant click pour sélectionner la macro désirée, il est ensuite possible de l'insérer à la position voulue dans le dessin. A cet instant, nous devrions nous trouver dans une situation similaire à celle montrée en figure 6.

Une chose que l'on peut remarquer tout de suite est que le transistor bipolaire sur la gauche n'est pas orienté de la façon correcte. Il suffit de faire click sur le bouton «Sélectionner», dans la barre des outils, sélectionner le transistor avec la souris et appuyer sur la touche [S]. Les éléments sélectionnés apparaîtront marqués en vert. Le résultat que l'on va obtenir devrait être similaire à celui montré en figure 7.

## 2. LE DESSIN AVEC FIDOCADJ

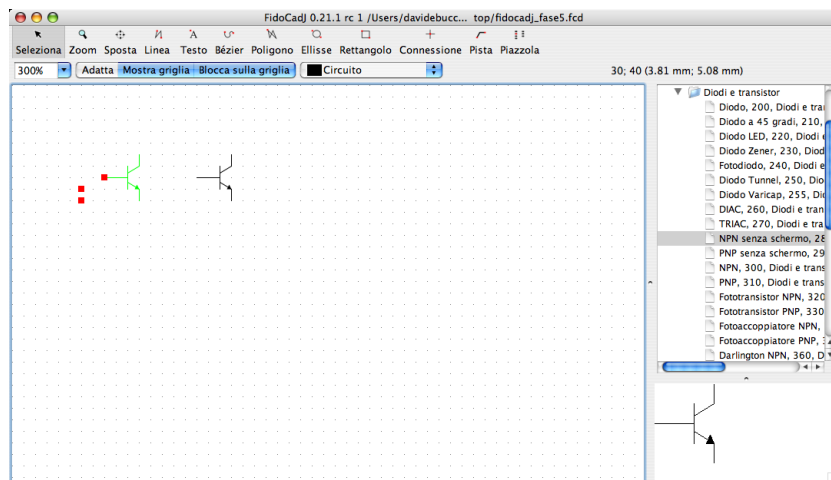


FIG. 6: Commençons à positionner les deux transistors.

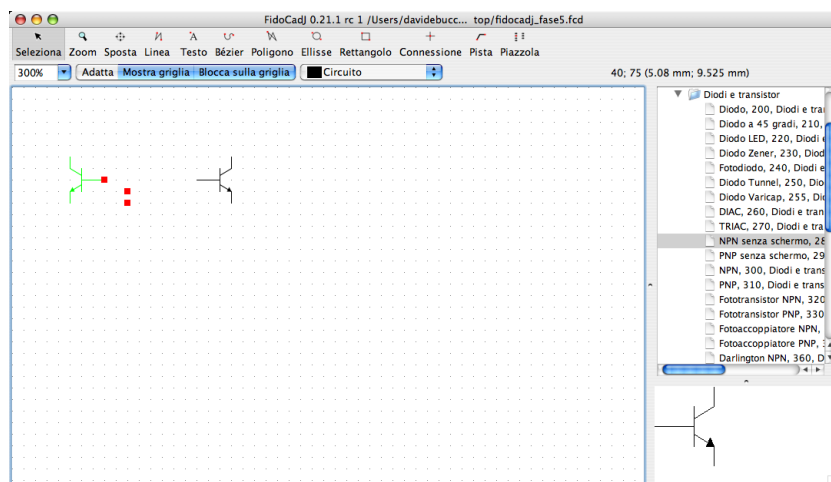


FIG. 7: Nous pouvons sélectionner le transistor sur la gauche et le retourner avec **S**.

## 2.2. Dessinons un simple schéma électrique

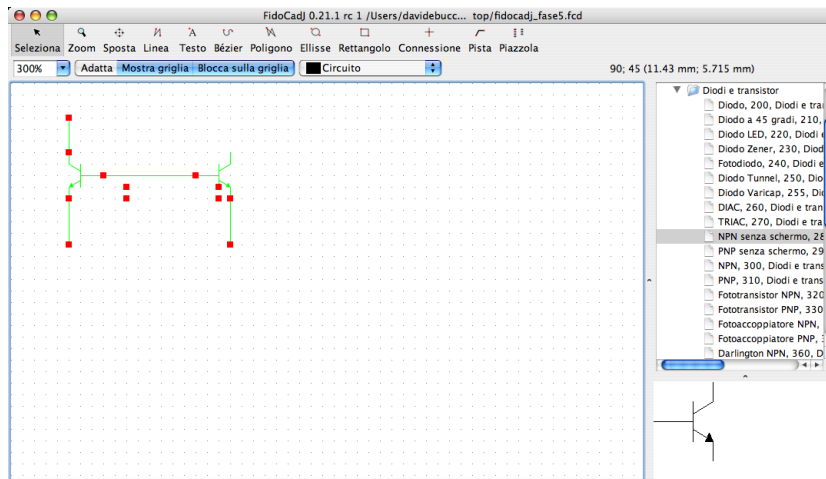


FIG. 8: Nous sommes trop en haut : nous pouvons tous sélectionner et déplacer vers le bas de la fenêtre.

En utilisant l'outil «ligne» dans la barre des outils, nous pouvons compléter quelques connexions électriques, jusqu'à nous apercevoir d'avoir commencé le dessin un peu trop près des bords de la surface de dessin. Pas de problèmes : nous allons nous placer en modalité de sélection et faire click en haut à gauche, en déplaçant la souris jusqu'à en bas à droite, toujours en maintenant enfoncée la touche. Un rectangle vert apparaîtra, qui nous indiquera que nous sommes en train de sélectionner tous les éléments à son intérieur. Nous allons tout sélectionner, comme montré en figure 8. A ce point, toujours en modalité sélection, il suffit de faire click sur un élément sélectionné quelconque et déplacer la souris, pour déplacer tous les éléments dans la position souhaitée.

Nous allons donc continuer en introduisant les parties manquantes du circuit, en particulier une résistance (Librairie standard/Composants discrets/Résistance), ainsi que le terminal d'alimentation positive (Librairie standard/Symboles de base/Terminal +). Ce dernier devra être fait pivoter pour lui donner l'orientation souhaitée. Il suffira de le sélectionner et appuyer sur la touche **R**, jusqu'à obtenir le résultat souhaité. A ce point, nous devrions avoir quelque chose de similaire au résultat montré en figure 9.

Ce qu'il manque maintenant est seulement le texte et la flèche qui représente le sens du courant. Pour cette dernière, il y a la macro «Flèche» dans «Librairie standard/Symboles de base». Pour le texte, il faut utiliser le bouton «Texte», dans la barre des commandes, et faire click sur le dessin à la position où le texte devra apparaître. Un texte «String» apparaîtra et pourra être changé en y cliquant avec la touche droite de la souris. Le nom et le modèle du transistor utilisé (en réalité, on utiliserait un couple de transistors sur le même substrat) sont spécifiés à l'intérieur des champs «Name» et «Value», que l'on obtient en faisant double click en modalité «Sélection» sur la macro.<sup>3</sup> Vous pouvez voir en figure 4 le résultat. Une dimension de la police adaptée pour travailler avec les schémas est de 4 unités en vertical et 3 en horizontal. Le circuit terminé est montré en figure 10.

<sup>3</sup> La possibilité d'associer un nom et une valeur à une macro, c'est à dire à un symbole d'un composant, représente une extension de FidoCadJ par rapport à FidoCad. Lire le paragraphe 3.4 pour avoir plus d'informations sur la compatibilité.

## 2. LE DESSIN AVEC FIDOCADJ

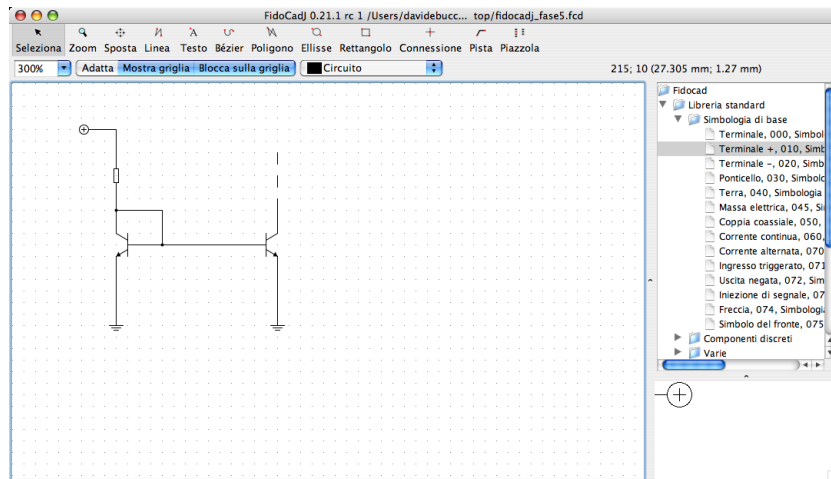


FIG. 9: Le circuit, presque terminé.

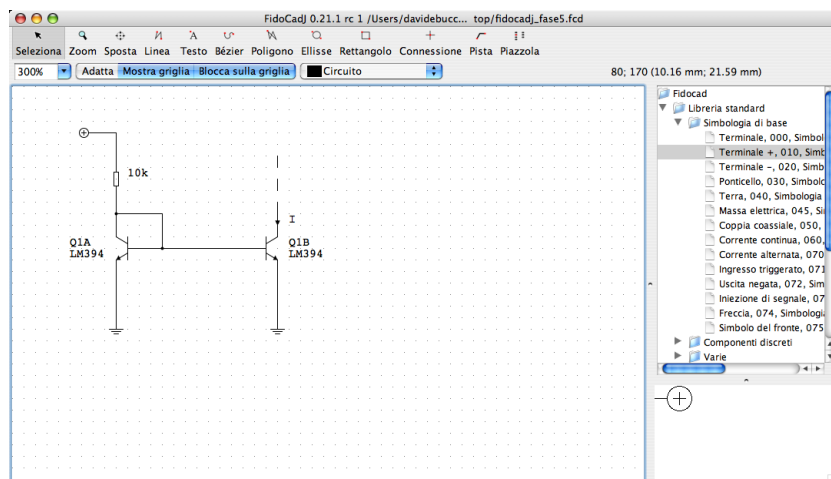


FIG. 10: Voici le circuit terminé, dans toute sa splendeur.

Pour les curieux, voici le code textuel qui décrit le circuit, et qui pourra être copié à l'intérieur d'un message dans un groupe de discussion, un message de courrier électronique, ou un forum. Pour le voir, il suffit de sélectionner «Texte du circuit», dans le menu «Circuit».

```
[FIDOCAD]
MC 95 65 0 0 280
FCJ
TY 115 60 4 3 0 0 0 * Q1B
TY 115 65 4 3 0 0 0 * LM394
MC 55 65 0 1 280
FCJ
TY 20 60 4 3 0 0 0 * Q1A
TY 20 65 4 3 0 0 0 * LM394
LI 55 65 95 65 0
LI 40 75 40 95 0
LI 110 75 110 95 0
LI 40 40 40 55 0
MC 40 30 0 0 115
LI 40 15 40 30 0
LI 30 15 40 15 0
MC 30 15 2 0 010
LI 40 50 60 50 0
LI 60 50 60 65 0
SA 60 65 0
SA 40 50 0
LI 110 45 110 55 0
LI 110 35 110 40 0
LI 110 25 110 30 0
MC 40 95 0 0 040
MC 110 95 0 0 040
TY 45 30 4 3 0 0 0 * 10 k
TY 115 50 4 3 0 0 0 * I
MC 110 50 1 0 074
```

Si vous voulez comprendre les détails de ce format, dans ce manuel il y a une description détaillée au chapitre 3.

Vous n'êtes pas forcés à utiliser la fenêtre «Texte du circuit» ; il suffit en fait de sélectionner la partie du circuit qui vous intéresse et la coller à l'intérieur du message : le code apparaîtra de façon automatique.

## 2.3 LES COUCHES (LAYER)

Une façon de se rendre compte de la fonction des couches est d'imaginer à un dessin fait sur des transparents pour rétro projecteur. Le résultat final est donné par une superposition de plusieurs transparents superposés. Chaque couche est distinguée par une couleur spécifique et on peut choisir si la représenter ou pas. Cette façon d'agir est commune à beaucoup d'outils de dessin techniques ou électroniques, car il permet de représenter facilement plusieurs parties du circuit qui seront superposés par exemple dans un circuit imprimé.

FidoCadJ permet d'utiliser 16 couches, numérotées de 0 à 15. La fonction de chaque couche est basé sur une convention et en particulier la couche 0 est utilisée pour les schémas électriques, la couche 1 pour les circuits, du côtés des soudures, la couche 2 pour le cuivre côté composants et la couche 3 pour la sérigraphie. Les couches qui restent ne sont pas associées à une fonction en particulier et peuvent être utilisées comme l'on souhaite. Le nom et la couleur de chaque couche

peuvent être spécifiés à travers le menu «Vue/Options couches». Dans le même menu, on peut aussi décider si une couche doit être montrée à l'écran, ou pendant l'impression.

L'ordre des couches est important. En particulier, les couches d'ordre plus faible seront dessinées en premier. Des dessins présents sur les couches successives pourront donc couvrir ce qui se trouve dans les couches d'ordre faible.

### 2.4 LA GRILLE

L'unité logique utilisée par FidoCadJ est de 5 mils (127 microns) et l'on ne peut pas avoir des demi-unités, dans le sens où les coordonnées de chaque élément doivent toujours être entières. Cela est suffisant à obtenir une résolution acceptable pour les schémas électriques et la plupart des circuits imprimés.

Pour nous rendre la vie plus facile, FidoCadJ peut utiliser une grille plus grossière, et faire en sorte que chaque opération faite avec la souris soit automatiquement alignée à cette grille. Pour cette raison, deux boutons «Montrer la grille» et «Bloquer sur la grille» sont présents. Ils permettent de montrer les points qui font partie de la grille, et d'activer ou pas la fonction d'alignement automatique. Le pas de la grille peut être sélectionné à l'intérieur de la fenêtre activée avec le menu «Vue/Options de dessin du circuit».

### 2.5 DESSINER UN SIMPLE CIRCUIT IMPRIMÉ

Pour prendre en main tous les concepts vus jusqu'à maintenant, la meilleure façon de procéder est de voir comment il faut procéder pour dessiner un circuit imprimé à l'aide de FidoCadJ.

À la différence d'autres programmes de CAO électrique qui sont sans doute très puissants, mais aussi plutôt difficiles à utiliser, FidoCadJ fournit en pratique une version électronique des vieux transférables R41. Bien évidemment, le fait de travailler à l'ordinateur permet de bénéficier de toute la flexibilité offerte par ce moyen.

Il faut néanmoins remarquer que le projet d'un circuit imprimé, surtout si ce dernier est complexe, n'est pas une tâche très simple. Ils existent, certes, des placeurs automatiques, ainsi que des routeurs automatiques et ils promettent des miracles dans les brochures publicitaires des fournisseurs. Il est cependant encore un fait que le routage est un travail qui ne peut pas se passer de l'intelligence et de l'expérience de l'opérateur. FidoCadJ fournit un éditeur rapide et immédiat pour dessiner des petits circuits imprimés à la portée d'un amateur. Nous allons voir rapidement comment il faut se comporter pour en obtenir un très simple, mais raisonnablement complet.

Ce que je conseille à qui désire effectuer ce travail est de se faire une idée suffisamment précise d'où placer les composants et les pistes principales avant de commencer. Le principe est de faire croiser le moins de connexions que possible.

Nous allons tricher, et nous allons commencer directement du résultat que nous souhaitons obtenir, montré en figure 11. Il s'agit d'un simple amplificateur à un étage, réalisé autour d'un transistor bipolaire (BC547 ou similaires) monté à émetteur commun. Pour comprendre la façon de travailler de FidoCadJ, il pourra être utile d'imaginer la plaque comme si elle était transparente, en la regardant du côté des composants. Pour cela, il est utile de travailler avec les sérigraphies, qui nous aideront à

*Le lecteur se fasse courage : un peu de raisonnements faits avec une feuille de brouillon, un stylo (et beaucoup de gommes) lui permettront à la fin d'avoir des idées plus précises avant de se mettre devant l'ordinateur et à la fin d'économiser du temps.*

## 2.5. Dessiner un simple circuit imprimé

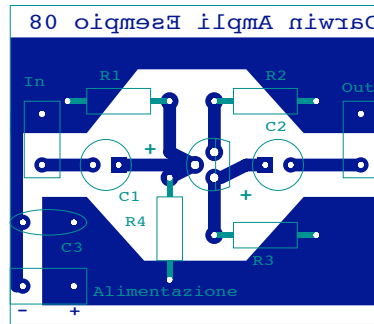


FIG. 11: Un très simple étage amplificateur à émetteur commun.

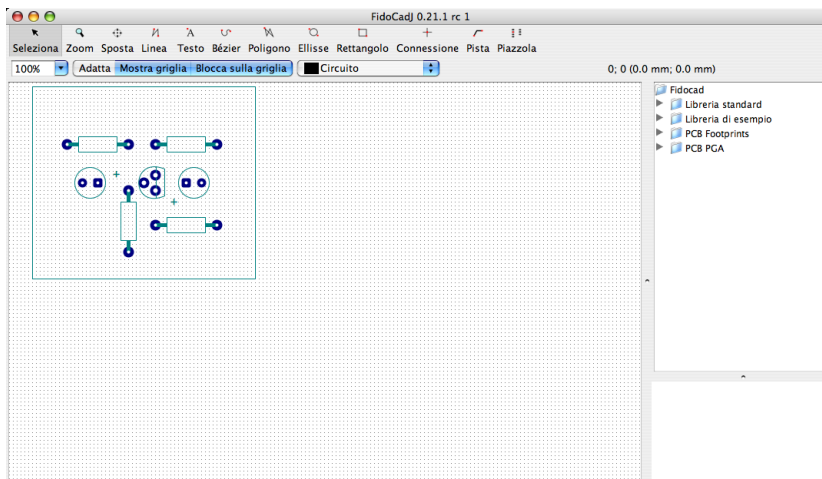


FIG. 12: Les composants sont placés sur la plaquette.

nous repérer, à partir du moment où il y aura plusieurs composants sur le circuit. Cela est un guide utile, même si souvent elle ne sera pas transférée sur la plaque lors d'une fabrication «maison».

La première chose que je conseille de faire est de disposer les composants, même de façon approximative. Dans notre cas, il faudra placer le transistor (bibliothèque «PCB footprints/Semiconducteurs 3 pattes/TO92»), les résistances (bibliothèque «PCB footprints/Résistances/Résistance 1/W 0,4 i»), les condensateurs électrochimiques («PCB footprints/Condensateurs électrochimiques/Vert. diam 5 pas 2,5»). Il peut être utile de marquer tout de suite la dimension totale souhaitée du circuit, en introduisant un rectangle vide sur la couche des sérigraphies (la 3). Pour effectuer cela, il suffit de se placer sur la couche souhaitée à travers la liste déroulante en haut à droite, et utiliser ensuite l'élément rectangle dans la barre des commandes.

Nous devrions obtenir plus ou moins le résultat montré en figure 12.

Nous allons donc définir les régions couvertes de cuivre, qui iront nous fournir l'alimentation positive et négative. Pour réaliser cela, il va falloir utiliser des polygones, en utilisant l'élément correspondant dans la barre des outils. Une fois rentrée chaque polygone, nous allons spécifier qu'il doit être plein, en faisant click droit sur une de ses côtés et en cochant l'option correspondante à l'intérieur de la fenêtre qui apparaîtra. Faites attention aussi à travailler sur la couche correcte, c'est à dire la 1 : c'est à dire le cuivre du côté des soudures. L'utilisation d'un polygone pour les plans d'alimentation peut être utile pour disposer

## 2. LE DESSIN AVEC FIDOCADJ

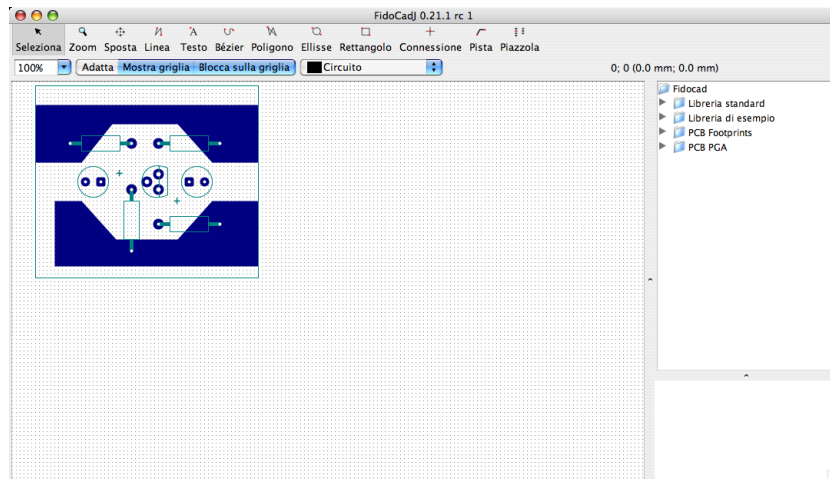


FIG. 13: Nous avons rajouté les régions de masse et d'alimentation positive à l'aide de deux polygones.

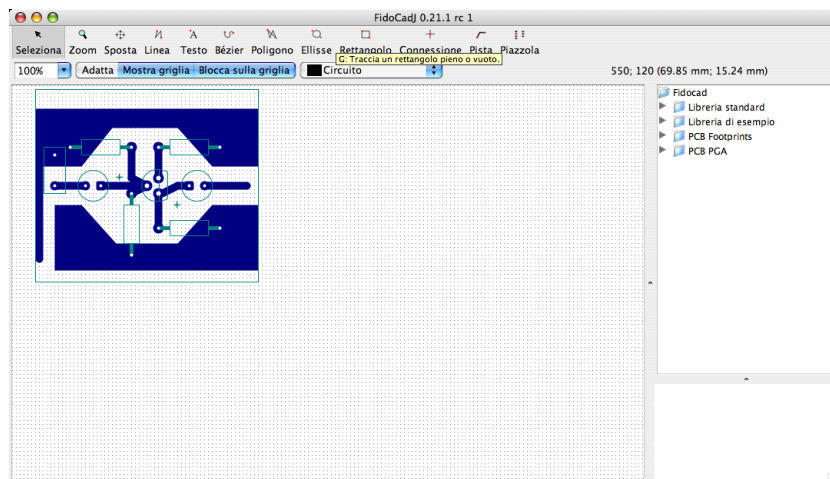


FIG. 14: Nous avons rajouté les connexions de masse et l'alimentation positive, à l'aide de deux polygones.

*Attention à la largeur des pistes : ce qui a l'air d'être une autoroute sur l'écran, se révélera dans la réalité une fine ligne prête à se décoller pendant la soudure.*

de connexions à basse impédance parasite. Avec un peu d'attention, normalement vous devriez avoir le résultat montré figure 13.

Il faudra ensuite compléter les connexions électriques, en utilisant les éléments «pistes c. i.» (pistes pour circuit imprimé). J'ai choisi pour ce circuit une largeur de 10 unités (1,27 mm), qui est une largeur plutôt commode par exemple pour rendre faciles les soudures.

Nous avons besoin de quelques connecteurs : pour pouvoir injecter le signal, ainsi que pour le récupérer et fournir l'alimentation au circuit. Nous allons utiliser une pastille prévue pour un condensateur au polyester, qui va probablement avoir la dimension correcte. FidoCadJ à la fin est une évolution des transférables Letraset. . .

Nous rajoutons aussi des repères pour l'alimentation + et - du côté cuivre, avec un condensateur céramique en parallèle. Nous allons aussi rajouter l'écriture sur la partie supérieure de la plaque. Pour écrire sur le côté cuivre, il faudra sélectionner la couche correspondante et écrire en miroir. Ceci est possible en accédant aux propriétés de l'élément, en faisant click avec la touche droite sur le texte à modifier. Faites quelques essais pour obtenir les dimensions optimales de la police. Pour commencer, il est utile de garder un rapport approché de 3/4



## 2.5. Dessiner un simple circuit imprimé

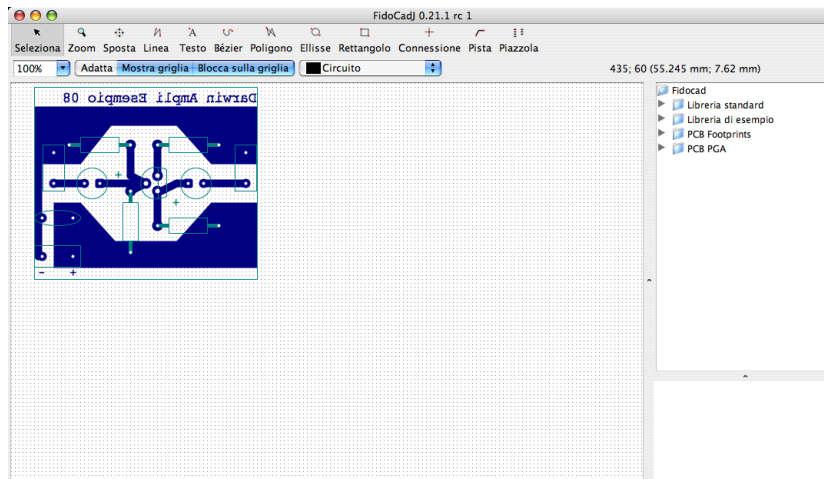


FIG. 15: Vioci le circuit imprimé, presque terminé.

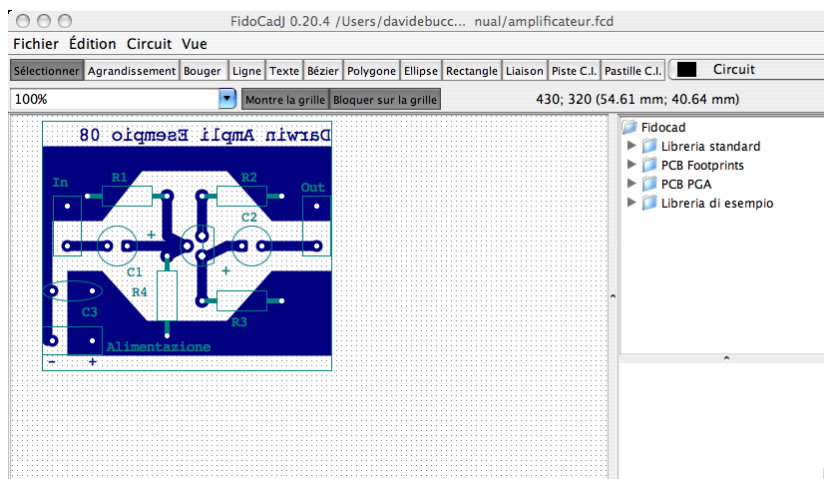


FIG. 16: Le travail terminé, avec toutes les sérigraphies.

entre les tailles horizontales et verticales des caractères. La figure 15 montre le résultat obtenu avec des tailles de 11 unités en horizontal et 18 en vertical.

À ce point, la seule chose qui manque est le texte avec les noms des composants, qui pourra être positionné sur la couche 3, qui est dédiée à la sérigraphie. Une image de FidoCadJ qui montre le circuit terminé est visible en figure 16.

Une fois le travail terminé, il faudra probablement imprimer le circuit sur calque, de façon à utiliser la photo incision, ou les méthodes de transfert direct du toner. Pour cette raison, seules les pistes de cuivre devront apparaître lors de l'impression. Pour obtenir ce résultat, il faudra en premier lieu cacher toutes les couches qui ne devront pas apparaître. Cela peut être fait à travers la fenêtre de dialogue qui apparaît en cliquant sur le menu «Vue/Options des couches». Dans notre cas, il suffira de rendre invisible la couche 3, qui présente les sérigraphies. Le logiciel ira montrer donc seulement le cuivre de côté des soudures.

Il faudra donc imprimer ce que l'on montre à l'écran (SANS l'adapter à la page, bien évidemment, de façon à respecter les dimensions choisies), et sélectionner l'impression en noir et blanc. Il pourra être

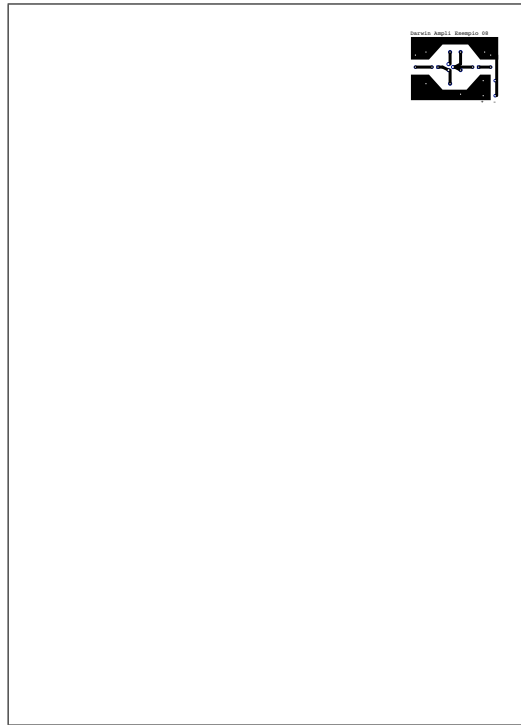


FIG. 17: Le circuit imprimé, tel qu'il apparaît imprimé en miroir sur une page en format ISO-UNI A4.

utile d'imprimer en miroir, si la technique que vous allez utiliser pour réaliser le circuit imprimé le demande.

Comme notre circuit imprimé est relativement petit, à grandeur naturelle il sera imprimé dans un coin d'une feuille en format standard ISO-UNI A4, comme l'on peut voir en figure 17.

Si quelqu'un est curieux, voici le code du circuit imprimé que l'on vient d'obtenir dans les exemples ci dessus :

```
[FIDOCAD]
TY 320 10 18 11 0 4 1 * Darwin Ampli Esemplio 08
TY 85 240 12 8 0 5 1 * +
TY 44 239 12 8 0 5 1 * -
PL 35 90 35 225 10 1
PL 55 130 95 130 10 1
PL 250 130 305 130 10 1
PL 215 130 230 130 10 1
PL 195 140 215 130 10 1
PL 115 130 175 130 10 1
MC 155 220 3 0 PCB.R01
MC 75 80 0 0 PCB.R01
MC 270 185 2 0 PCB.R01
MC 270 80 2 0 PCB.R01
MC 230 130 3 0 PCB.CE00
MC 115 130 1 0 PCB.CE00
MC 40 175 0 0 PCB.CC50
PL 190 80 190 120 10 1
PL 190 140 190 185 10 1
PL 155 80 155 120 10 1
PL 155 120 175 130 10 1
PL 155 140 175 130 10 1
PP 30 30 30 105 90 105 130 55 215 55 260 105 320 105
    320 30 1
```

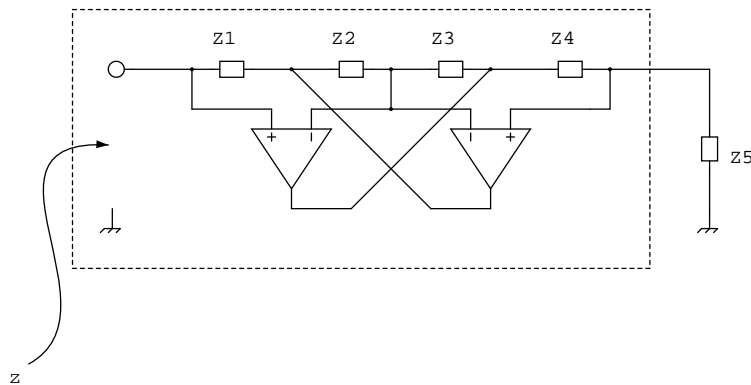


FIG. 18: Un schéma électrique d'un GIC dans lequel on a utilisé quelques extensions de FidoCadJ.

```

PP 320 240 320 155 260 155 215 205 135 205 90 155 55
    155 55 240 1
MC 190 120 0 0 PCB.T092
MC 305 90 1 0 PCB.CPBX352
MC 55 90 1 0 PCB.CPBX352
MC 80 225 2 0 PCB.CPBX352
TY 290 65 12 8 0 0 3 * Out
TY 40 60 12 8 0 0 3 * In
TY 95 225 12 8 0 0 3 * Alimentazione
TY 70 190 12 8 0 0 3 * C3
TY 230 95 12 8 0 0 3 * C2
TY 115 150 12 8 0 0 3 * C1
TY 120 170 12 8 0 0 3 * R4
TY 220 200 12 8 0 0 3 * R3
TY 230 55 12 8 0 0 3 * R2
TY 100 55 12 8 0 0 3 * R1
RV 30 5 320 255 3

```

## 2.6 FRECCHE E STILI DI TRATTO

À partir de la version 0.23 de FidoCadJ, j'ai rajouté des possibilités vis à vis du dessin. Vous pouvez maintenant rajouter des flèches sur les extrémités des segments, ainsi que des courbes de Bézier. FidoCadJ offre maintenant la possibilité de choisir de quel côté les dessiner, ainsi que avec quel style parmi ceux disponibles. J'ai rajouté aussi des styles de trait interrompus, pour faciliter le dessin de schémas techniques. La figure 18 montre un exemple dans lequel on a dessiné un étage analogique (un GIC) à l'intérieur d'un rectangle avec trait interrompu et avec une flèche au bout d'un tracé de Bézier. En faisant double click sur cet élément, la fenêtre des paramètres qui est montrée est visible figure 19.

Vous pouvez remarquer que l'option «Arrow at start» (flèche au début) est activée. Le logiciel dessinera une flèche au bout de la courbe, orientée de façon correcte. Vous pouvez choisir plusieurs styles de flèche et de trait, dans les listes actionnables à côté des points «Arrow style» et «Dash style». Faites quelques tests pour comprendre comment fonctionne le mécanisme.

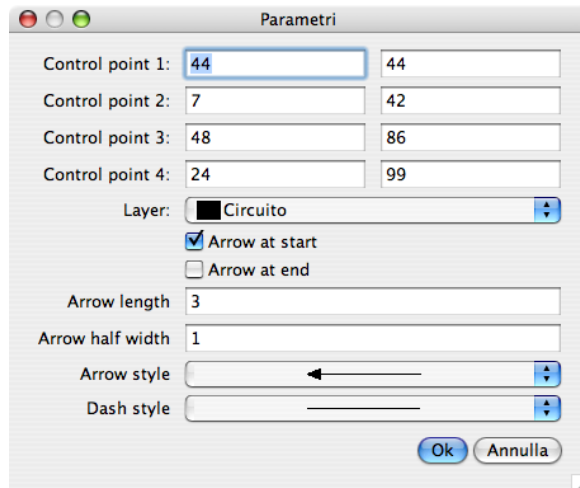


FIG. 19: La fenêtre des paramètres d'un tracé de Bézier, dans un dessin FidoCadJ.

La possibilité de choisir un type de trait et d'ajouter des flèches aux lignes n'est pas prévue dans le format FidoCad originel. Cela implique que malheureusement la compatibilité à l'arrière vers FidoCad pour Windows n'est pas assurée. Quand on choisit d'utiliser une extension FidoCadJ, il faut qu'il soit bien clair ce que l'on fait. Si l'on a besoin de maintenir une compatibilité avec des utilisateurs de FidoCad et non pas FidoCadJ, on peut activer l'option «Modalité de compatibilité avec FidoCad» à l'intérieur de l'onglet «Extension FidoCadJ» dans la fenêtre «Préférences de FidoCadJ», avant de commencer le travail. De cette façon, on ne pourra pas introduire des éléments graphiques qui n'ont pas été prévus par FidoCad et l'on obtiendra des dessins parfaitement compatibles avec ce dernier. Pour plus d'informations, consulter le paragraphe 3.4.

## 2.7 EXPORTATION

Personnellement, une des choses que je trouve les plus intéressantes et utiles de FidoCadJ est la possibilité de créer des petits schémas pour une utilisation typographique. Pour cette raison, j'ai essayé de permettre et faciliter la mise au point des exportations des dessins dans des formats graphiques divers et variés.

Pour réaliser une exportation du dessin courant, il suffit de sélectionner l'entrée «Exportation» dans le menu «Fichier». Le tableau 2 montre une liste des formats graphiques qui sont actuellement disponibles pour l'exportation. .

Pour chaque format, le tableau spécifie si le format est vectoriel ou à matrice de points. Quand il est possible, il est préférable d'utiliser systématiquement des formats vectoriels pour l'exportation, de façon à obtenir les résultats les meilleurs dans toutes les situations.<sup>4</sup>

Pour les formats à matrice de points, il est peut être utile d'utiliser l'option «anticrénelage», pour limiter l'effet visuel «en échelle» des lignes et des éléments du dessin. Les informations sur la résolution, ainsi que l'option anticrénelage ne sont pas utilisées lors de l'exportation dans un format vectoriel.

*Un format vectoriel  
mémoire les  
éléments du dessin.  
Un format à matrice  
de points travaille sur  
une matrice de pixels  
à une résolution  
donnée.*

<sup>4</sup> La structure du code de FidoCadJ permet de rajouter avec simplicité des formats vectoriels pour l'exportation. Si vous voulez me donner un coup de main. .

L'option «Noir et blanc» permet d'imprimer tous les éléments visibles seulement en noir plein. Cette fonctionnalité est utile par exemple quand on est en train de préparer des calques pour photo incision. Il faut par contre dire quelque mots sur l'exportation en PDF faite par FidoCadJ. Le logiciel ne peut pas inclure des polices à l'intérieur du fichier généré. Cela veut dire que l'on obtiendra des fichiers identiques à l'original seulement pour les dessins qui utilisent les polices standard type 1 à 14 points :

- Courier : Police avec empattements à espacement fixe. On fait automatiquement la conversion en Courier des écritures faites avec Courier New.
- Times : Police avec empattements à espacement variable. Le logiciel convertit en Times aussi les éléments en Times New Roman, ainsi qu'en Times Roman.
- Helvetica : Police sans empattements à espacement variable. On y convertit aussi les éléments faits en Arial.
- Symbol : contient plusieurs symboles.

Entre autre, ces polices sont pratiquement toujours disponibles presque partout. C'est donc une très bonne idée de les utiliser de façon systématique.

## 2.8 OPTIONS DE LA LIGNE DE COMMANDE

FidoCadJ est distribué sous la forme d'un fichier `.jar`, c'est à dire un archive Java<sup>5</sup> Dans beaucoup de systèmes d'exploitation, pour lancer l'application il est probablement suffisant de faire double click sur le fichier, à condition d'avoir une version de Java installée sur la machine. Dans la terminologie Sun, ce qu'il faut avoir est le JRE, c'est à dire «Java Runtime Environment» : tout ce qu'il faut pour exécuter un logiciel écrit en Java (mais pas pour l'écrire ; pour cela il faut le SDK...).

Dans quelques situations, il peut être utile de faire démarrer FidoCadJ depuis l'invite des commandes (le «terminal» des systèmes Unix, ou le «MS-DOS prompt» pour les systèmes Windows). Pour cela, il suffit d'utiliser la commande `java`, avec l'option `-jar` :

```
java -jar fidocadj.jar
```

Si un fichier FidoCad est spécifiée dans la commande, FidoCadJ va essayer de l'ouvrir. Par exemple, observons la commande (valable dans un système Unix) :

```
java -jar fidocadj.jar ~/FidoCadJ/test.fcd
```

FidoCadJ sera démarré et essayera d'ouvrir le fichier `~/FidoCadJ/test.fcd` (à condition que ce dernier existe!).

Une autre possibilité intéressante, même si à la rigueur il s'agit plutôt d'une caractéristique de Java plus que de FidoCadJ lui même, est qu'il est possible de modifier l'apparence du logiciel en changeant ce qui en jargon Java l'on appelle le «look and feel».

Vous pouvez néanmoins jouer avec l'aspect que vous préférez depuis l'invite des commandes, sans modifier une seule ligne de code. Voici quelque chose que les utilisateurs Linux probablement vont apprécier : l'aspect GTK+ :

<sup>5</sup> Sauf dans la version pour Apple Macintosh, pour laquelle il y a une application distribuée sous la forme traditionnelle.

Format	Commentaires
JPG	Format à matrice de points très diffusé. La compression utilisée est avec pertes et ce format n'est pas adapté à l'exportation de schémas comme ceux produits par FidoCadJ.
PNG	Format compressé, à matrice de point. En manque d'un format vectoriel, ce format est celui le plus adapté à être utilisé pour des schémas et des graphiques.
SVG	Format vectoriel standard proposé par le W3C. Certains navigateurs Internet (comme les versions récentes de Safari) permettent de le visualiser en une page web. Bon format à utiliser pour des graphiques et des schémas, il permet d'utiliser des logiciels tels qu'Inkscape pour retoucher les dessins. Actuellement, des limitations existent pour le texte ayant une orientation différente de l'usuelle.
EPS	Format vectoriel Postscript encapsulé, très utilisé par les logiciels de graphique professionnelle. Ce format a été utilisé pour obtenir la figure 11, à la page 13, en passant par une conversion en PDF, car ce manuel a été produit avec PDF $\LaTeX$ .
PGF	Format vectoriel prêt à être utilisé directement à l'intérieur d'un fichier source $\LaTeX$ , à l'aide du paquet <i>pgf</i> , mis à disposition dans l'archive CTAN. Cette modalité d'exportation a été conçue pour exporter les schémas électriques et fournit un code qui peut être modifié à la main. Les textes ne sont pas interprétés : cela permet d'introduire du code $\LaTeX$ directement à l'intérieur du dessin. Cette technique a été utilisée dans ce manuel pour obtenir la figure 5, à la page 7.
SCR	FidoCadJ, à partir de la version 0.21, permet d'exporter un dessin vers un script interprétable par le logiciel CadSoft Eagle. Pour se servir de cette possibilité, il faut copier dans le répertoire <code>1br</code> d'Eagle le fichier <code>FidoCadJLIB.1br</code> , téléchargeable depuis le site de FidoCadJ. Actuellement, l'exportation est possible seulement pour les schémas électriques qui adoptent les symboles les plus utilisés. Les pastilles et les pistes pour les circuits imprimés ne sont pas exportés.
PDF	Le format Portable Document Format, de Adobe. Voir le texte pour ce qui concerne les limitations sur les polices à utiliser.

TAB. 2: Liste des formats d'exportation disponibles avec FidoCadJ.

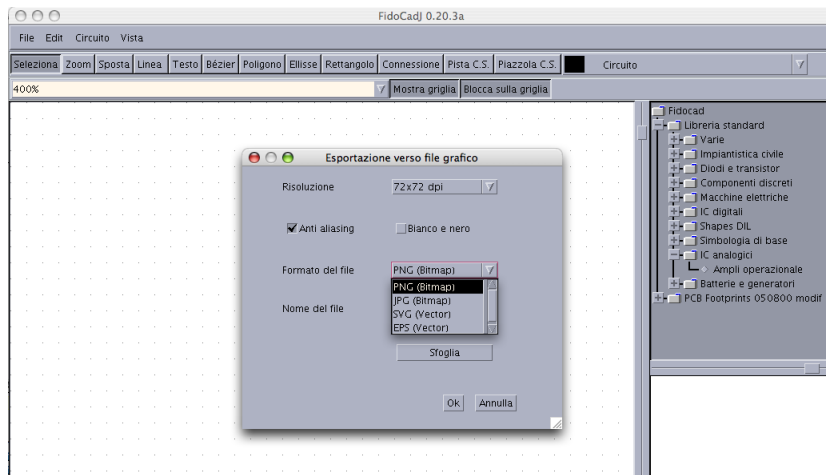


FIG. 20: L'aspect de FidoCadJ sous MacOSX, en utilisant le «look and feel» Motif.

```
java -Dswing.defaultlaf=com.sun.java.swing.plaf.gtk.
    GTKLookAndFeel -jar fidocadj.jar
```

Ou aussi le classique «look and feel» Motif, qui est montré dans la figure 20 :

```
java -Dswing.defaultlaf=com.sun.java.swing.plaf.motif.
    MotifLookAndFeel -jar fidocadj.jar
```

Il est clair que les commandes que l'on vient de voir doivent être utilisés depuis l'invite des commandes, une fois que l'on se trouve dans le même répertoire où l'on a placé le fichier `fidocadj.jar`, en écrivant tout sur la même ligne.

## 2.9 GESTION DES LIBRAIRIES

FidoCadJ permet de spécifier un répertoire à l'intérieur duquel on trouve des fichiers de librairie (avec l'extension `.fcl`) qui seront chargés à chaque démarrage du logiciel. Pour faire cela, il suffit d'ouvrir la fenêtre «Vue/Options» et spécifier le répertoire dans la ligne «Sélectionner le répertoire librairies». Si FidoCadJ trouve un fichier nommé `FCDstdlib.fcl`, il chargera la librairie standard depuis ce fichier. S'il trouve par contre un fichier nommé `PCB.fcl`, il l'utilisera pour la librairie des pastilles pour circuits imprimés.<sup>6</sup> Les librairies contenues dans d'autres fichiers avec extension `.fcl` seront rajoutés à celles standard. Tout cela est fait au démarrage de FidoCadJ, donc n'oubliez pas de redémarrer le logiciel si vous y changez quelque chose.

À partir de la version 0.23, grâce à l'accord de Roby IZ1CYN, j'ai pu inclure la librairie IHRaM 2.1 directement à l'intérieur de l'archive contenant FidoCadJ. Cela pare que parmi les nombreuses librairies mises à point par les utilisateurs elle m'a apparue être la plus complète et rationnelle. Par contre, si dans le répertoire contenant les librairies externes au logiciel (comme décrit auparavant) on trouve un fichier appelé `IHRaM.FCL`, ce dernier sera chargé en mémoire à la place de la version contenue dans FidoCadJ.

<sup>6</sup> Faites attention à la casse des noms de ces fichiers, si votre système d'exploitation fait une différence entre les lettres minuscules et les majuscules.

# 3

## LE FORMAT DES DESSINS, DES MACROS ET DES LIBRAIRIES FIDOCAD

---

Dans ce chapitre, nous allons décrire en détail le format utilisé par FidoCad et de conséquence par FidoCadJ pour mémoriser les dessins. Il s'agit d'un très simple format de texte qui a l'avantage d'être compact et efficace. Comme ce format n'a jamais été décrit de façon complète, je propose ici un résumé de tout ce que j'ai appris pendant l'étude que j'ai faite.

### 3.1 DESCRIPTION DE L'EN-TÊTE

Tous les fichiers qui contiennent un dessin en format FidoCad doivent commencer avec l'en-tête standard `[FIDOCAD]`. Un logiciel peut reconnaître donc la présence de commandes FidoCad en reconnaissant cette en-tête. En réalité, FidoCadJ est beaucoup plus tolérant du FidoCad original et reconnaît et interprète correctement un fichier qui ne présente pas l'en-tête. Même des lignes de commande au milieu du texte sont reconnues correctement, à moins que le nombre de ligne consécutives non lisibles par le logiciel ne dépasse pas une certaine valeur (environ une centaine). Cela évite que FidoCadJ travaille inutilement, et pendant longtemps, si l'utilisateur essaye d'ouvrir un gros fichier binaire.

### 3.2 LE SYSTÈME DE COORDONNÉES

FidoCadJ travaille avec un système de coordonnées très simple. En pratique, le logiciel dessine sur une surface très grande (presque sans limites), identifiée par contre par des coordonnées entières et positives. La largeur de chaque unité en  $x$  et en  $y$  est fixée à  $127\ \mu\text{m}$ , valeur qui dans la pratique s'est avéré suffisante pour décrire les empreintes SMD, sans par contre être trop petite pour les utilisation les plus courantes.

Le logiciel FidoCad originel travaillait dans deux modalités différentes : PCB (circuit imprimé) et schéma électrique. En FidoCadJ, cette différence est beaucoup moins marquée et apparaît seulement au moment d'imprimer un dessin : il est en fait judicieux de modifier la dimension à l'impression un schéma électrique pour s'adapter à la surface disponible, tandis qu'il faudra respecter scrupuleusement les dimensions d'un circuit imprimé.

### 3.3 ÉLÉMENTS DE DESSIN

En FidoCadJ, il y a 11 primitives de dessin qui sont les suivantes :

- Ligne
- Rectangle plein ou vide
- Texte simple (obsolète)
- Texte avancé
- Polygone plein ou vide
- Ellipse plein ou vide
- Courbe de Bézier
- Connexion électrique
- Pastille pour circuit imprimé



a	Flèche
0	aucune
1	au début
2	à la fin
3	aux deux côtés

TAB. 3: Présence des flèches aux extrêmes d'un segment ou d'un tracé de Bézier, selon la valeur du code a.

b	Style de la flèche
0	flèche pleine et normale
1	flèche pleine avec trait
2	flèche vide
3	flèche vide avec trait

TAB. 4: Style des flèches, selon la valeur du code b.

- Piste pour circuit imprimé
- Macro

Nous allons analyser en détail le format utilisé pour chaque élément. En général, chaque élément est identifié par une commande suivie par une série de paramètres (normalement, des entiers, ou des chaînes de caractères) sur la même ligne, séparés d'un espace.

#### Ligne (segment)

L'élément ligne est identifié par la commande **LI**, suivie des coordonnées initiales, finales ainsi que du numéro de couche à utiliser :

```
LI x1 y1 x2 y2 l
```

Le point  $(x_1, y_1)$  représente les coordonnées initiales,  $(x_2, y_2)$  celles finales et l est le numéro du calque, compris entre 0 et 15.

**Extension FidoCadJ** : à partir de la version 0.23 de FidoCadJ, **LI** peut être suivi à la ligne suivante par une extension :

```
FCJ a b c d e
```

où a est un entier qui représente la présence ou l'absence de flèches aux extrêmes du segment (voir le tableau 3), b est un entier qui identifie le style de la flèche à utiliser (voir le tableau 4). Les paramètres c et d fournissent respectivement la longueur totale et la demi-ouverture de la flèche, tandis que e est le style du trait (interrompu ou pas).

#### Rectangle plein ou vide

Un rectangle plein ou vide est indiqué par les commandes **RP** (plein) et **RV** (vide), suivies par les coordonnées des points d'une diagonale du rectangle à tracer, ainsi que par le numéro de la couche à utiliser.

```
RP x1 y1 x2 y2 l
RV x1 y1 x2 y2 l
```

les coordonnées  $(x_1, y_1)$  représentent le premier point de la diagonale,  $(x_2, y_2)$  le deuxième point et l est le numéro du calque, compris entre 0 et 15.

*Les mathématiciens trouveraient probablement plus approprié le terme «segment».*

Bit	Poids	Fonction
0	1	Texte en noir
2	4	Texte en miroir

TAB. 5: Fonctions des bits dans le style du texte.

**Extension FidoCadJ** : à partir de la version 0.23 de FidoCadJ, [RP](#) et [RV](#) peuvent être suivis par un'extension dans la ligne suivante :

```
FCJ e
```

où *e* est un entier qui fournit le style à utiliser pour le trait.

#### Texte simple (obsolète)

Le texte simple a été le premier élément de texte mis à disposition par les premières versions de FidoCad.

Cet élément de dessin a été considéré obsolète par Lorenzo Lutti, le créateur de FidoCad. FidoCadJ se comporte de la même façon et, même si des dessins qui utilisent cet élément sont visualisés correctement, l'utilisateur ne peut pas l'introduire à partir de la barre des commandes. FidoCadJ convertira dans la mémoire cette commande dans l'élément «texte avancé» et il utilisera la commande [TY](#) lors de la sauvegarde du fichier.

La commande utilisée est [TE](#) et le format est le suivant :

```
TE x1 y1 texte à écrire
```

le point  $(x_1, y_1)$  spécifie où les caractères «texte à écrire» seront positionnés. Nous pouvons remarquer que l'indication du calque n'apparaît pas. FidoCadJ traitera cet objet comme appartenant au calque zéro (circuit).

#### Texte avancé

L'élément texte avancé permet une flexibilité bien plus grande par rapport à l'élément texte simple que l'on vient de voir.

Cet élément est identifié par la commande [TY](#), suivie par plusieurs paramètres qui spécifient l'orientation du texte (texte pivoté, ou en miroir), ainsi que les dimensions en *x* et en *y* de la police utilisée. Comme il commence à y avoir pas mal d'informations à fournir, la ligne de commande est plutôt articulée :

```
TY x1 y1 sy sx a s l f texte à écrire
```

le point  $(x_1, y_1)$  représente l'endroit où le texte «texte à écrire» sera positionné. Les valeurs de  $s_y$  et  $s_x$  indiquent la dimension verticale et horizontale du texte en unités logiques. J'ai choisi de faire en sorte que FidoCadJ calcule la dimension verticale du texte à partir de celle horizontale, et qu'il applique des déformations à la polices seulement si strictement nécessaire.

La rotation du texte est spécifiée par le terme *a*, exprimé en degrés, tandis que la valeur de *s* détermine le style du texte, selon le tableau 5. La couche utilisée est représentée par le terme *l*, tandis que *f* indique la police à utiliser, ou il est une étoile, pour indiquer l'utilisation de la police standard, qui est Courier New. Si le nom de la police contient des espaces, ceux derniers sont remplacés par le symbole +.

La longueur maximale du texte est d'environ quatre vingt mots. Le compte est effectué en mots, et non pas en caractères, car dans la structure interne du logiciel le mots sont séparés pendant que le dessin est chargé en mémoire.

#### *Polygone plein ou vide*

Un polygone plein ou vide est indiqué respectivement par les commandes **PP** et **PV**, suivies par les coordonnées des sommets, qui définissent le polygone, et la couche.

```
PP x1 y1 x2 y2 ... l
PV x1 y1 x2 y2 ... l
```

les points  $(x_1, y_1)$ ,  $(x_2, y_2)$ ... sont les coordonnées des sommets qui définissent le polygone et  $l$  est le numéro du calque, indiqué avec une chiffre comprise entre 0 et 15. La longueur totale de la ligne est donc variable et dépend du nombre de sommets qui sont présents. Le nombre maximum de sommets disponibles est fixé à 20 de façon arbitraire, pour éviter des lignes trop longues.

**Extension FidoCadJ** : à partir de la version 0.23 de FidoCadJ, **PP** et **PV** peuvent être suivis par un'extension dans la ligne suivante :

```
FCJ e
```

où  $e$  est un entier qui fournit le style à utiliser pour le trait.

#### *Ellipse pleine ou vide*

Un'ellipse pleine ou vide est indiquée respectivement par les commandes **EP** et **EV**, suivies par les coordonnées des deux extrêmes d'une diagonale et du numéro de la couche :

```
EP x1 y1 x2 y2 l
EV x1 y1 x2 y2 l
```

Le point  $(x_1, y_1)$  représente le premier point de la diagonale et  $(x_2, y_2)$  le deuxième. Le terme  $l$  est le numéro de la couche, identifiée par un entier compris entre 0 et 15.

**Extension FidoCadJ** : à partir de la version 0.23 de FidoCadJ, **EP** et **EV** peuvent être suivis par un'extension dans la ligne suivante :

```
FCJ e
```

où  $e$  est un entier qui fournit le style à utiliser pour le trait.

#### *Courbe de Bézier*

Une courbe de Bézier, dans sa variante cubique, est identifiée par quatre points qui sont donc requis par la commande **BE** :

```
BE x1 y1 x2 y2 x3 y3 x4 y4 l
```

Les points  $P_1 \equiv (x_1, y_1)$ ,  $P_2 \equiv (x_2, y_2)$ ,  $P_3 \equiv (x_3, y_3)$  et  $P_4 \equiv (x_4, y_4)$  sont les quatre points de contrôle de la courbe de Bézier, tandis que  $l$  est le numéro de la couche, identifiée par un entier compris entre 0 et 15. Une fois identifié les quatre points que l'on vient de voir, la courbe est donnée par l'équation suivante, qui effectue une combinaison entre les différents points :

$$B(t) = (1-t)^3 P_1 + 3t(1-t)^2 P_2 + 3t^2(1-t) P_3 + t^3 P_4 \quad (3.1)$$

où le terme  $t \in [0, 1]$  est un paramètre.

**Extension FidoCadJ** : à partir de la version 0.23 de FidoCadJ, **BE** peut être suivi à la ligne suivante par une extension :

```
FCJ a b c d e
```

où  $a$  est un entier qui représente la présence ou l'absence de flèches aux extrêmes du segment (voir le tableau 3),  $b$  est un entier qui identifie le style de la flèche à utiliser (voir le tableau 4). Les paramètres  $c$  et  $d$  fournissent respectivement la longueur totale et la demi-ouverture de la flèche, tandis que  $e$  est le style du trait (interrompu ou pas).

#### Connexion électrique

L'élément connexion électrique est tout simplement un petit cercle plein de dimension constante et il est utilisé pour représenter une connexion entre deux fils dans un schéma électrique. Il est caractérisé par la commande **SA** et nécessite uniquement des coordonnées, ainsi que la couche :

```
SA x1 y1 l
```

Avec FidoCadJ, le diamètre du cercle est fixé à 2 unités logiques et ne peut pas être modifié par l'utilisateur.

#### Pastille pour circuit imprimé

Une pastille pour circuit imprimé est identifiée par la commande **PA** et est caractérisée par son style (rond, rectangulaire, rectangulaire avec les coins arrondis), ainsi que par la dimension du trou interne :

```
PA x1 y1 dx dy si st l
```

Le point  $(x_1, y_1)$  va nous donner la position de la pastille,  $d_x$  et  $d_y$  sont la largeur et l'hauteur de la pastille. La valeur de  $s_i$  sera le diamètre interne du trou et  $s_t$  le style :

- 0 pastille ovale
- 1 pastille circulaire
- 2 pastille rectangulaire avec les coins arrondis

La valeur de  $l$  indique le numéro de la couche où la pastille sera tracée.

#### Piste pour circuit imprimé

La piste pour circuit imprimé est en gros rien d'autre qu'un segment, pour lequel on peut spécifier la largeur. Les extrêmes du segment sont toujours arrondis, pour simplifier la connexion avec les autres pistes du circuit. La commande à utiliser est **PL**, avec le format qui suit :

```
PL x1 y1 x2 y2 di l
```

La piste est tracée entre les points  $(x_1, y_1)$  et  $(x_2, y_2)$ , avec l'épaisseur  $d_i$ , sur la couche  $l$ .

#### Macro

Une macro est un dessin ou un symbole présent à l'intérieur d'une librairie. Très typiquement, les symboles électriques les plus utilisés sont représentés de cette façon. La commande à utiliser pour introduire une macro est **MC**, en utilisant la structure suivante :

```
MC x1 y1 o m n
```

La macro est dessinée aux coordonnées  $(x_1, y_1)$ , avec un angle donnée par la valeur de  $o$  (multipliée fois  $90^\circ$ ). Si  $m$  vaut 1, la macro est tracée en miroir. Le dernier paramètre,  $n$  est le nom de la macro, spécifié sous la forme `librairie.clé`. Il y a une exception importante à cette règle pour les éléments de la librairie standard (et seulement pour eux !), où la macro est appelée seulement par sa clé.

**Extension FidoCadJ** : à partir de la version 0.21 de FidoCadJ, `MC` peut être suivi par un code `FCJ` à la ligne suivante :

```
[FIDOCAD]
MC 40 30 0 0 080
FCJ
TY 50 35 4 3 0 0 0 * R1
TY 50 40 4 3 0 0 0 * 47k
```

La présence de cette commande indique que la définition de la macro n'est pas terminée, mais qu'il y a des informations en plus concernant le nom et la valeur de la macro, spécifiés à l'aide de deux commandes `TY`. Actuellement (0.23), FidoCadJ prend en compte les coordonnées, le texte ainsi que la police des caractères utilisée.

Cette façon de procéder a l'avantage que si un fichier contenant des extensions est lu par FidoCad, ce dernier fournira un message d'erreur, mais lira correctement le schéma (voir le paragraphe 3.4).

### 3.4 EXTENSIONS DE FIDOCADJ

À partir de la version 0.21, FidoCadJ a introduit des extensions au format FidoCad originel. Dans le code, ces extensions sont identifiées par la commande `FCJ`. La présence de cette commande indique que l'instruction précédente n'est pas terminée, mais qu'il faut spécifier davantage d'informations.

FidoCadJ introduit à partir de la version 0.23 des extensions plus importantes, mais dispose par contre d'une modalité de «compatibilité avec FidoCad» qui est contrôlable d'après l'onglet «Extensions FidoCadJ» à l'intérieur de la fenêtre «Préférences». Si cette modalité est activée, toutes les extensions propres à FidoCadJ sont désactivées et le dessin sera parfaitement compatible avec FidoCad pour Windows. Dans le cas contraire, FidoCad sera encore capable de lire les schémas produits par FidoCadJ (bien qu'il fournira un message d'erreur), mais les informations en plus seront perdues.

Le résultat dépendra du contexte : une ligne dessinée à trait interrompu sera rendue comme une ligne continue. Les flèches qui étaient associées aux extrémités ne seront pas dessinées. Par contre, le texte associé à une macro (nom et valeur) sera affiché correctement. La figure 21 devrait résumer ce que l'on obtient lorsque l'on lit avec FidoCad le code utilisé pour dessiner la figure 18. FidoCad sera un peu énervé à cause des commandes qui ne reconnaîtra pas, il y a des détails qui manquent, mais le dessin demeure compréhensible.

### 3.5 TOLÉRANCE AUX ERREURS DE SYNTAXE

FidoCadJ a été écrit pour être particulièrement tolérant vis à vis d'erreurs, ou à des commandes incomplètes qui sont fournies au logiciel. Clairement, à moins que vous ayez une boule de cristal USB, le logiciel

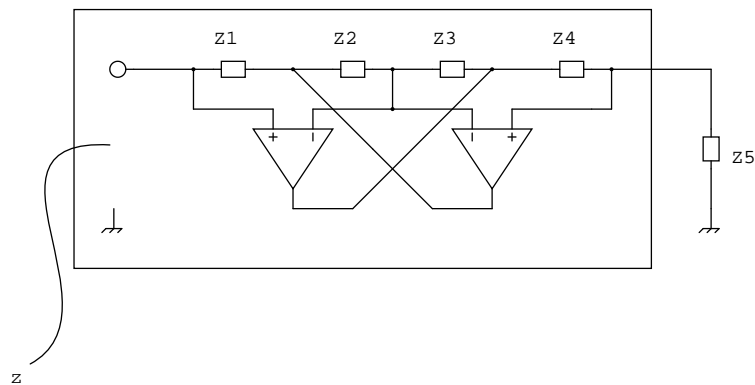


FIG. 21: La figure 18 comme apparaîtrait si montrée avec FidoCad.

ne pourra pas corriger les erreurs et il se limitera à sauter les lignes qui posent problèmes ou sont incomplètes.

Une exception à ce comportement est due au fait que, pour des raisons de compatibilité avec les toutes premières versions de FidoCad, certains éléments peuvent être spécifiés en omettant la couche à utiliser. Il seront dans ce cas considérés comme faisant partie de la couche 0, destinée au dessin des schémas électriques.

### 3.6 LE FORMAT DES LIBRAIRIES

La structure d'un fichier de librairie est très simple :

```
[FIDOLIB Librairie de base]
{Syboles de base}
[000 Terminal]
LI 100 100 102 100
EV 102 98 106 102
[010 Terminal +]
LI 100 100 102 100
EV 102 98 106 102
LI 103 100 105 100
LI 104 99 104 101
[020 Terminal -]
LI 100 100 102 100
EV 102 98 106 102
LI 103 100 105 100
...
```

La première ligne indique, entre crochets, le nom de la librairie (après le texte FIDOLIB). La deuxième ligne fournit, entre accolades, la catégorie de la librairie à l'intérieur de laquelle nous allons organiser les macros qui vont suivre dans le fichier.

Chaque macro est composée par un en-tête (entre crochets), ainsi qu'une suite de commandes de dessin. L'en-tête est composé par la clé (qui *doit* être unique dans chaque librairie), ainsi que d'une description. La clé est utilisée à l'intérieur du code FidoCad, tandis que la description servira à guider l'utilisateur dans le choix pendant l'utilisation du logiciel. Les commandes de dessin ne sont rien d'autre que des dessins FidoCad, autour du point (100, 100), qui est utilisé comme origine. Cette origine sera ensuite utilisée comme point de base pour

le positionnement de la macro. La macro sera utilisé à l'aide de la commande MC, avec le code «librairie.macro».

Nul n'empêche d'utiliser une macro à l'intérieur d'une autre macro. Ce qu'il faut éviter est par contre la récursion, c'est à dire de définir une macro qui fait appel à soi même.

### 3.7 LIBRAIRIES STANDARD

FidoCadJ contient à son intérieur plusieurs librairies fournies traditionnellement avec FidoCad. En particulier, il s'agit de la librairie standard et de la librairie dédiée aux symboles des composants pour les circuits imprimés(PCB).

Il est néanmoins possible de substituer le contenu des librairies internes à FidoCadJ, en spécifiant au logiciel (menu "Vue/Options/- Répertoire librairies") un répertoire contenant les librairies à charger. Si un fichier nommé `FCDstdlib.fcl` est présent dans ce répertoire, son contenu sera utilisé à la place de la librairie standard. Si par contre un fichier nommé `PCB.fcl` est présent, son contenu sera utilisé à la place de la librairie contenant les symboles pour les circuits imprimés. Librairies contenues dans des fichiers avec un nom différent, mais toujours avec l'extension `fcl` seront chargées à côté des librairies standard.

# 4

## CONCLUSION

---

Dans ce manuel, nous avons vu comment utiliser FidoCadJ pour dessiner un schéma électrique et un simple circuit imprimé. A ce point, le lecteur devrait posséder tous les éléments nécessaires pour utiliser de façon créative FidoCadJ pour ses exigences.

Il ne faut pas croire que FidoCadJ est un instrument dédié seulement à l'électronique. En préparant des bibliothèques spécifiques, il devient possible de l'utiliser pour une variété très grande de dessins bi-dimensionnels.

Il est très important que les utilisateurs puissent communiquer avec moi (en particulier, pour savoir dans quelle direction continuer le développement). N'hésitez donc pas à me contacter ([davbucci@tiscali.it](mailto:davbucci@tiscali.it), mais sans aucun type de fichier joint!).





## EXTENSIONS SPÉCIFIQUES AUX SYSTÈMES D'EXPLOITATION

---

### A.1 MACOSX

#### A.1.1 Extensions

Une des critiques les plus courantes faites aux premières versions de FidoCadJ de la part des utilisateurs Macintosh (comme moi, du reste) concerne la mauvaise intégration de ce logiciel sous MacOSX. À partir de la version 0.21.1, FidoCadJ a fait des efforts spécifiques pour s'intégrer avec l'apparence et le fonctionnement des applications natives. Pour cette raison, certains détails sur le fonctionnement du logiciel sont légèrement différents lorsque FidoCadJ se rend compte d'être en train de tourner sur une plate-forme Apple :

- FidoCadJ utilise par défaut le look and feel Quaqua<sup>1</sup> quand il est fait démarrer à partir de l'application complète (FidoCadJ.app, pas donc juste le fichier fidocadj.jar). Quaqua pourrait introduire des ralentissements sur des ordinateurs pas très récents. Pour cette raison, il est possible de le désactiver à partir de la fenêtre Préférences.
- La barre du menu se trouve à sa place, c'est à dire dans la partie haute de l'écran.
- Les entrées «Préférences...» et «À propos de FidoCadJ» se trouvent à leur place, c'est à dire à l'intérieur du menu «FidoCadJ».
- Le logiciel déclare au système d'exploitation sa disponibilité à ouvrir les fichiers de type .fcd. Il y associe aussi une icône évocatrice.

#### A.1.2 Comment télécharger et exécuter FidoCadJ sous MacOSX

FidoCadJ peut fonctionner avec une version de MacOSX supérieure ou égale à la version 10.3.9 (Panther). La raison est que FidoCadJ demande une version de Java supérieure à la 1.4. Apple fournit normalement tout ce qu'il faut pour exécuter une application Java à l'intérieur de son système d'exploitation vedette.

Même s'il est tout à fait possible de télécharger et utiliser le fichier `fidocadj.jar` comme l'on fait avec d'autres systèmes d'exploitation, il est plutôt indiqué d'utiliser l'application que j'ai mise à disposition exprès pour ce système. Tout est tout à fait identique aux applications natives ; il suffit de télécharger l'image disque qui contient le logiciel à l'adresse suivant : <http://sourceforge.net/projects/fidocadj/files/FidoCadJ.dmg/download>, ouvrir l'image disque et déplacer FidoCadJ.app à l'intérieur du dossier Applications, où il sera disponible comme tout autre logiciel installé dans le système. Pour désinstaller FidoCadJ, il sera suffisant de déplacer FidoCadJ.app dans la corbeille.

---

<sup>1</sup> Quaqua est disponible à l'adresse : <http://www.randelshofer.ch/quaqua/>

## A. EXTENSIONS SPÉCIFIQUES AUX SYSTÈMES D'EXPLOITATION

### A.2 LINUX

#### A.2.1 Extensions

Sous Linux, le logiciel ne prévoit pas d'extensions spécifiques.

#### A.2.2 Comment télécharger et exécuter FidoCadJ sous un système Linux

*par Roby Pozzato IZ1CYN*

Presupposé : avoir installé le JRE 6 par Sun et/ou OpenJDK 6 JRE (ou les versions précédentes compatibles avec ce qui demande FidoCadJ). Dans le paragraphe A.2.3, nous allons décrire comment installer le logiciel en utilisant seulement des commandes à fournir dans une invite de commandes. Dans le paragraphe A.2.4, nous allons par contre voir ce qu'il faut faire dans un système pourvu d'une interface graphique.

#### A.2.3 Sur n'importe quel système, depuis l'invite des commandes

Nous allons télécharger le logiciel en utilisant la commande `wget` :

```
$ wget http://davbucci.chez-alice.fr/elettronica/
  fidocadj/fidocadj.jar
--23:51:22--  http://davbucci.chez-alice.fr/elettronica
               /fidocadj/fidocadj.jar
               => 'fidocadj.jar'
Risoluzione di davbucci.chez-alice.fr in corso...
  212.27.63.127
Connessione a davbucci.chez-alice.fr
  |212.27.63.127:80... connesso.
HTTP richiesta inviata, aspetto la risposta... 200 OK
Lunghezza: 276,694 (270K) [application/x-java-archive]

100% [=====>] 276,694
              71.75K/s      ETA 00:00

23:51:26 (71.59 KB/s) - "fidocadj.jar" salvato
               [276694/276694]
```

Nous allons créer un répertoire (mais tout d'abord devenons super-utilisateurs avec la commande `su` ou `sudo -s`)

```
# mkdir /usr/bin/fidocadj
```

... et nous allons y déplacer le fichier que l'on vient de télécharger (substituez `<user>` avec le nom de l'utilisateur qui a téléchargé le fichier) :

```
# mv /home/<user>/fidocadj.jar /usr/bin/fidocadj
```

Allons donc rendre exécutable le fichier :

```
# chmod +x /usr/bin/fidocadj/fidocadj.jar
```

Et n'oublions pas de redevenir des utilisateurs normaux !

```
# exit
```

Maintenant, nous pouvons lancer le logiciel :

```
$ /usr/bin/fidocadj/fidocadj.jar
```

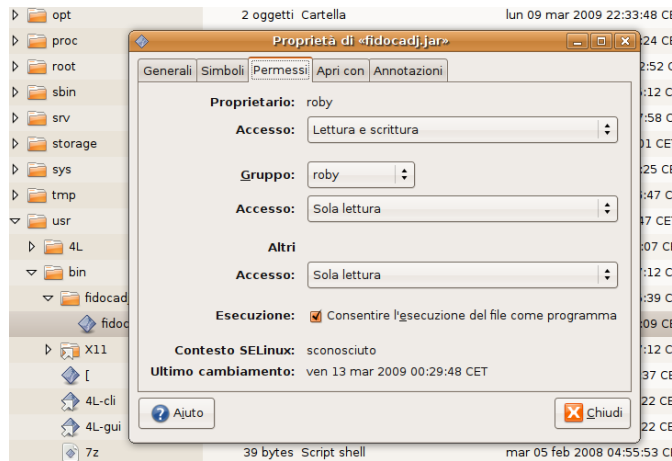


FIG. 22: La fenêtre de gestion des permissions, dans la distribution Ubuntu 8.04.

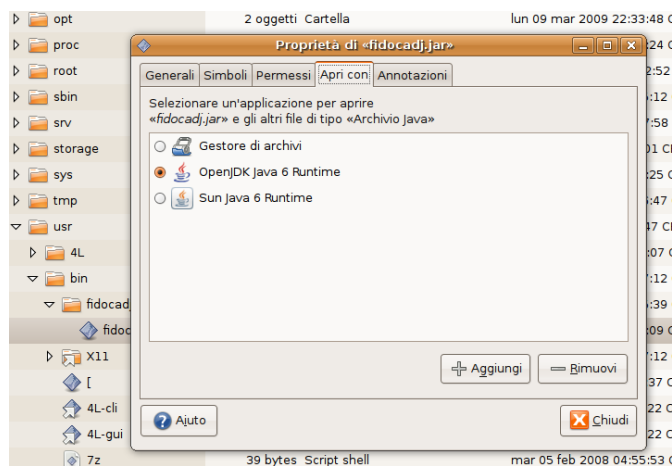


FIG. 23: Sélectionner l'exécution de la machine virtuelle Java, dans la distribution Ubuntu 8.04.

#### A.2.4 Sur un système graphique

Dans notre exemple, nous allons décrire ce qu'il faut faire pour une distribution Ubuntu 8.04. De toute façon, pour des versions précédentes ou successives (ou bien d'autres systèmes), ne changent que des détails :

- Nous pouvons télécharger le fichier avec le navigateur Internet, voire avec des outils tels que Gwget.
- Nous allons démarrer notre outil de gestion de fichier (Nautilus, Konqueror, etc.) en tant que super-utilisateur. S'il n'y a pas une option exprès pour effectuer cela dans les menus, il suffit de le faire démarrer à partir de l'invite des commandes avec `sudo nautilus`. Nous allons créer le répertoire `/usr/bin/fidocadj` et nous pouvons y déplacer le fichier que nous venons de télécharger (probablement, il se trouve en `/home/<user>/`)
- En faisant click droit sur le fichier, dans la fenêtre nous allons sélectionner l'onglet «Permissions». Nous pouvons sélectionner l'option «Permettre l'exécution du fichier en tant que logiciel», tel que montré dans la figure 22.

- En sélectionnant l’onglet «Ouvrir avec», nous pouvons sélectionner l’option «OpenJDK Java 6 Runtime», ou «Sun Java 6 Runtime», comme montré figure 23.<sup>2</sup>
- Faisons click sur «Fermer» et nous sommes prêts à exécuter FidoCadJ : un double click sur l’exécutable, ou nous pouvons le rajouter au menu ; la commande à utiliser est tout simplement : `/usr/bin/fidocadj/fidocadj.jar`.

### A.3 WINDOWS

À partir de la version 0.23, si FidoCadJ est fait tourner sous Windows, on utilise par défaut le «Look and Feel» destiné à ce système d’exploitation.

---

<sup>2</sup> Note. On m’a signalé qu’en utilisant le runtime OpenJDK, on risque d’avoir des problèmes lors de l’impression sur des imprimantes qui utilisent des pilotes CUPS. Si vous avez des problèmes, faites des tests sur d’autres systèmes telles que Sun JRE 1.4.2, si disponible.

## INDEX

---

- [+](#), [24](#)
- [écrire en miroir](#), [14](#)
- [élément](#), [4](#), [23](#)
- [élément de dessin](#), [4](#)
- [étoile](#), [24](#)
- 
- [A4](#), [16](#)
- [Arial](#), [19](#)
- 
- [Bézier](#), [6](#), [17](#), [22](#), [25](#)
- [barre des commandes](#), [4](#)
- [barre des outils](#), [9](#)
- [BE](#), [25](#), [26](#)
- [bouge](#), [6](#)
- [boule de cristal USB](#), [27](#)
- 
- [CadSoft](#), [20](#)
- [CadSoft Eagle](#), [1](#)
- [circuit](#), [24](#)
- [circuit imprimé](#), [11](#), [12](#), [15](#), [16](#)
  - [empreintes](#), [4](#)
  - [PCB](#), [22](#)
  - [SMD](#), [22](#)
- [Cocoa](#), [2](#)
- [code](#), [11](#)
- [compression avec pertes](#), [20](#)
- [connexion](#), [6](#), [22](#), [26](#)
- [couche](#), [13](#), [28](#)
- [couches](#), [11](#)
- [couleur](#), [11](#)
- [Courier](#), [19](#)
- [Courier New](#), [19](#), [24](#)
- [courrier électronique](#), [11](#)
- [crochets](#), [28](#)
- [croiser les connexions](#), [12](#)
- 
- [dessin vectoriel](#), [4](#)
- 
- [Eagle](#), [20](#)
- [ellipse](#), [6](#), [22](#), [25](#)
- [en tête](#), [28](#)
- [en-tête](#), [22](#)
- [EP](#), [25](#)
- [EPS](#), [1](#), [20](#)
- [EV](#), [25](#)
- [exportation](#), [18](#)
- [extension FidoCadJ](#), [27](#)
- [extensions FidoCadJ](#), [27](#)
- 
- [FCJ](#), [27](#)
- [FidoCad](#), [1](#), [2](#), [4](#), [9](#), [18](#), [22](#), [24](#),  
[27–29](#)
- 
- [FidoCadJ](#), [9](#)
- [FidoCadJ.app](#), [31](#)
- [fidocadj.jar](#), [31](#)
- [FIDOLIB](#), [28](#)
- [FidoReadJ](#), [2](#)
- [flèche](#), [17](#)
- [format à matrice de points](#), [18](#)
- [format vectoriel](#), [18](#)
- [forum](#), [11](#)
- 
- [GIC](#), [17](#)
- [grille](#), [12](#)
- [groupe de discussion](#), [v](#), [11](#)
- [GTK+](#), [19](#)
- 
- [Helvetica](#), [19](#)
- 
- [impression](#), [15](#)
- [Inkscape](#), [20](#)
- [interprète](#), [2](#)
- [it.hobby.elettronica](#), [v](#), [2](#)
- [it.hobby.fai-da-te](#), [2](#)
- 
- [jar](#), [19](#)
- [Java](#), [1](#), [2](#), [19](#)
- [JPG](#), [20](#)
- [JRE](#), [19](#), [32](#)
- 
- [L<sup>A</sup>T<sub>E</sub>X](#), [20](#)
- [LI](#), [23](#)
- [librairie](#), [21](#)
- [librairie PCB](#), [29](#)
- [librairie standard](#), [7](#), [29](#)
- [ligne](#), [6](#), [22](#), [23](#)
- [ligne de commande](#), [19](#)
- [Linux](#), [1](#), [19](#), [32](#)
- [longueur maximale du texte](#), [25](#)
- [look & feel](#), [19](#)
- [Lorenzo Lutti](#), [1](#), [24](#)
- 
- [Macintosh](#), [5](#), [19](#)
- [MacOSX](#), [2](#), [4](#), [5](#), [21](#), [31](#)
- [macro](#), [1](#), [4](#), [7](#), [23](#), [26](#)
- [MC](#), [26](#), [27](#), [29](#)
- [Metal](#), [4](#), [5](#)
- [Motif](#), [21](#)
- [MS-DOS prompt](#), [19](#)
- 
- [newsgroup](#), [1](#)
- [nombre maximum de sommets](#),  
[25](#)
- 
- [obsolète](#), [24](#)

- OpenJDK, 32
- PA, 26
- Panther, 31
- paramètres, 23
- pastille c. i., 6
- pastille pour c. i., 22, 26
- PDF, 1, 20
- PDFLATEX, 20
- PGF, 20
- photo incision, 15
- piste c. i., 6, 14
- piste per c. i., 23
- piste pour c. i., 26
- PL, 26
- placeur automatique, 12
- PNG, 20
- polices, 19, 20
- polygone, 6, 13, 14, 22, 25
- polygones, 14
- Postscript, 20
- PP, 25
- programmation à objets, 2
- PV, 25
- Quaqu, 31
- réursion, 29
- résistance, 9
- recherche dans les librairies, 4
- rectangle, 6, 13, 22, 23
- routeur automatique, 12
- RP, 23, 24
- RV, 23, 24
- sélection, 4
- sérigraphie, 11, 12
- SA, 26
- Safari, 20
- schéma électrique, 22
- schémas électriques, 11
- SCR, 20
- SDK, 19
- segment, 23
- selection, 6
- soudures, 11
- style du texte, 24
- Sun, 4, 19, 32
- super-utilisateur, 32
- SVG, 20
- Symbol, 19
- système de coordonnées, 22
- tailles des caractères, 15
- TE, 24
- terminal, 19
- Texte, 9
- texte, 6, 22
- texte avancé, 24
- texte en miroir, 24
- texte pivoté, 24
- texte simple, 24
- Times, 19
- Times New Roman, 19
- Times Roman, 19
- tolérance aux fautes, 27
- trait, 17
- trait interrompu, 17
- transférables Letraset, 14
- transférables R41, 12
- transfert direct du toner, 15
- transistor, 7
- TY, 24
- Ubuntu, 33
- unité logique, 12
- unités logiques, 24
- Unix, 19
- Usenet, 1
- Windows, 1, 2, 19
- WInE, 1
- zoom, 6

Ce manuel a été rédigé en utilisant `pdfLATEX` sous MacOSX, en adoptant la classe `classicthesis`. Le code a été mis en page en utilisant le paquet `listings`. Le paquet `pgf` a été utilisé pour la figure 5. Les paquets sont disponibles dans l'archive CTAN.

Ce travail a été composé avec la police *Palatino*, de Hermann Zapf.