Mediaware Review

The Apple Media Kit Version 7.7

L. F. (Len) Proctor, Editor

System Information and Requirements

Apple Programmers Development Association (APDA)
Apple Computer, Inc.
Order: l-800-637-651 1
Fax: 716-871-6511
Applelink: APDA
Price Range: $l,OOO- $2,000

System Requirements:

- Macintosh IIci or later
- 8 MB of RAM memory
- At least 40 MB hard disk drive (although larger drives are recommended)
- System 7.1 or later
- QuickTimeTM, version 1.6.1 or later
- CD-ROM optional

Program Description

The Apple Media Kit is a cross-platform, multimedia authoring tool. The Media Tool component facilitates the assembly of sound, picture, computer graphics, text and movie files. The AMT Programming Environment is an object-oriented programming language that enables a programmer to add interactivity to a project. The Kit contains Macintosh installation disks, Windows installation disks, documentation, a demo CD containing an electronic version of the Programming Environment documentation, sample Mac and Windows projects and a demonstration version of the Media Tool.

Basic Functions

Media Tool projects use an icon map to structure the user path and the order in which screens are displayed. A library folder is used to hold all the media components that are used to construct the screen. When a completed Media Tool project is saved, the description of objects and their actions can be

output to the AMT Programming Environment as a text file. In text form, the code can be edited and enhanced with the Apple Media Language or other standard programming language such as Pascal or C. Conditional branching based on user input can only be achieved within the programming environment.

Documentation

Documentation for the Media Tool has two forms. Getting Started is a tutorial guide that explains the basic operation of the program and the User's Guide provides a reference type, feature by feature explanation. The Programming Environment has both a User's Guide and a Reference Guide. An electronically searchable copy of the programming documentation is also available on the companion CD-ROM. For the Media Tool, the Getting Started Guide presumes that the user has an extensive working knowledge of Macintosh operating conventions. The Programming Environment presumes a basic knowledge of the Macintosh Programmer's Workshop (MPW) shell, programming in C, object-oriented programming, Macintosh programming fundamentals and the Media Tool.

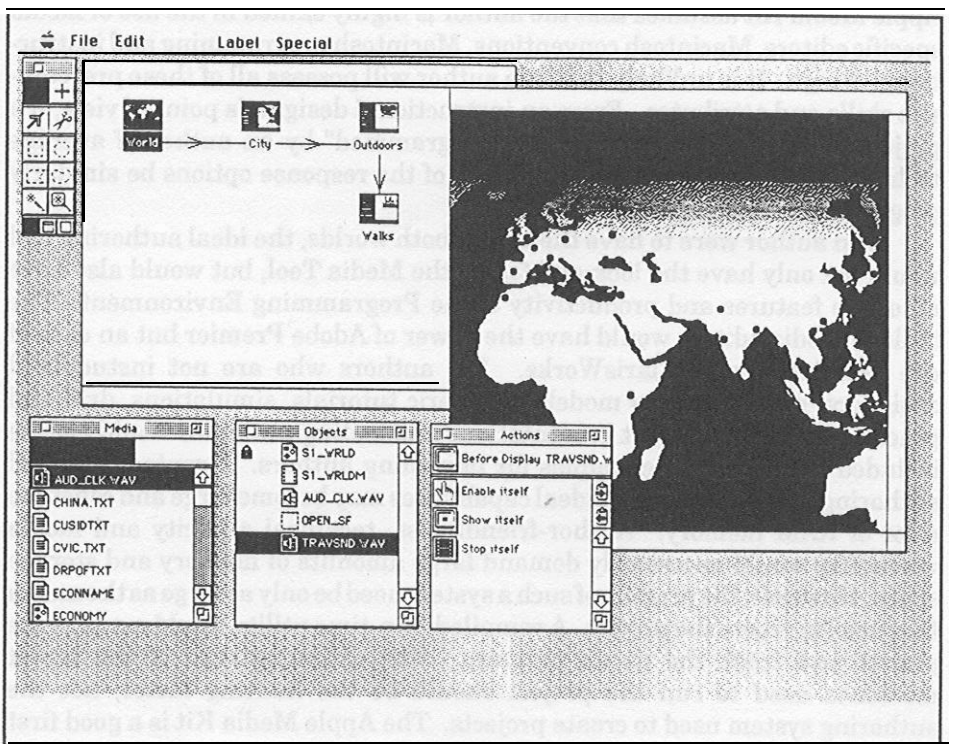
Critique and Recommendations

The Media Tool functions as a media assembly tool because it does not contain any resident media editors. Editing sound, text, picture or movie files has to be done by jumping out of the Media Tool to an appropriate outside editor. In order to use external media editors while the Media Tool is being used, there must be enough RAM memory available to handle each task. Second, if cross platform development is anticipated then MS-DOS file naming conventions must be used when the final editing is done, otherwise, the project may not run properly in a DOS or WINDOWS environment. The necessity of having to buy external editors adds to the cost of the program. Depending of the quality of subsidiary media editors selected, the cost will vary from a few dollars for shareware products to several hundred dollars for full-featured commercial editing programs.

The Media Tool requires a minimum of 3 MB on a hard drive. The documentation suggests that a minimum of 4 MB of RAM memory be exclusively dedicated to AMT, but recommends that the minimum be increased to 8 MB. Given that system files can easily consume another 2 to 3 MB, it is easy to see that if any multi-tasking is required, a minimum of 16 MB of RAM memory will be required for even a modest sized project. Second, because sound, still picture and video files are notorious for their ability to fill up hard disk space, the larger the hard drive available the better. Memory requirements for the Programming Environment are similar. The important point to note here is that while the program will run on smaller machines like the Macintosh IIci, using equipment with limited RAM memory and disk storage capabilities will very quickly limit the size and complexity of projects that can be developed.

Authors who want to start producing projects quickly will find the icon map feature very valuable because it makes thumbnails of all the project screens visible and clearly delineates the paths used to connect these screens. The map feature together with the use of library folder to hold the media components facilitates the rapid prototyping of a project and permits each member of a multi-member development team to input media files. One point to keep in mind however is that if the project contains more that fifty thumbnails, the scrolling speed of the map window becomes very slow. Even though thumbnails can be hidden to increase navigation speed, they still occupy 5 KB of memory. Project editing and revision is easy because as each screen or screen component is drafted, crafted and polished it can be added to the project by simply giving it the same name as the rough draft and adding the finished version to the library after the rough draft has been deleted.

User navigation through a project authored with the Media Tool is achieved by creating hot spots on menus, text elements or symbols. Like HyperCard, the hot spots can be linked to any screen or screens that are specified by the designer in the project map. Navigation commands are activated by the user when the mouse is clicked or moved, or when screens are opened and closed. No author programming, coding or scripting is required to specify navigation paths. The project author simply chooses an object and selects what action is to occur and when from a menu of choices (see illustration).



Once the project producer becomes familiar with the choices available under the menu items, tangible products can be created with a minimal investment of time in learning how to use the features of the Media Tool. Second, by limiting user input to click object responses AMT learning time can also be reduced. On the other hand, if the author wishes to use other forms of user response such as text input or to specify conditional branching based on user input, he or she has no choice but to revert to learning how to program a Macintosh. Unless the author is already an accomplished Macintosh programmer, his/her AMT learning time will quickly expand.

The response time of an AMT project to user is dependent on the complexity of the screen being assembled. This may vary from a few seconds to many seconds. Project response time can be enhanced by compiling code and preloadingpicture and movie files. However, running a project stored on a CD-ROM instead of a hard drive may cancel any gains in response time made by using these techniques. If the performance of the project at run time is judged to be too slow, the designer should consider dividing complex screens into two or more screens. There is a direct relationship between response time and object presentation. Hence, fewer objects on a screen at any one time will result in an increase in the speed of operation.

In conclusion, version 1.1 of the Apple Media Kit is one of the better choices of assembly tools that a multimedia author can select for simple project production. For more complex projects requiring higher levels of interactivity, Apple Media Kit assumes that the author is highly skilled in the use of media specific editors, Macintosh conventions, Macintosh programming and instructional design. It is unlikely that one author will possess all of these prerequisite skills and attributes. From an instructional designer's point of view, if a project navigation hot spot can be "programmed" by an author of average technical skill, why can't the remainder of the response options be similarly programmed from Media Tool menus?

If an author were to have the best of both worlds, the ideal authoring tool would not only have the look and feel of the Media Tool, but would also have all of the features and productivity of the Programming Environment. The built-in media editors would have the power of Adobe Premier but an ease of use associated with ClarisWorks. For authors who are not instructional designers, representative models of generic tutorials, simulations, drill and practice exercises, content presentations and testing examples would also be included in the Kit as examples for beginning authors. Granted, an ideal authoring environment with ideal capabilities may become large and consume a lot of RAM memory. Author-friendliness, technical capacity and media versatility would necessarily demand large amounts of memory and storage space. However, the product of such a system need be only as large as the media files used to create the project. A compiled, run-time utility would manage the project and track the user's progress. The characteristics of the target machines used to run the project should be the limiting factor, not the authoring system used to create projects. The Apple Media Kit is a good first

step in this direction. Hopefully, as development continues on this product, it will continue to evolve in an author-friendly direction and eventually have all the capabilities suggested for an ideal authoring system.

EDITOR

L.F. (Len) Proctor is Associate Professor of Curriculum Studies, University of Saskatchewan, Saskatoon, SK S7N OWO.