
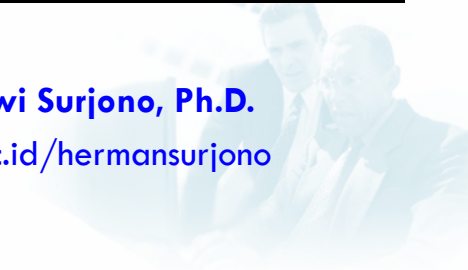



Pengantar Pemrograman untuk Aplikasi Pembelajaran

Prof. Herman Dwi Surjono, Ph.D.
<http://blog.uny.ac.id/hermansurjono>



Methods to Integrate Media



- **Programming**
 - Languages specify how the media is presented and user interactions carried out.
 - Requires command of the language.
 - Is time consuming.
- **Authoring**
 - Applications specially designed to integrate and present media elements.
 - Developers can concentrate on design, interactivity, and functionality of the project.






Authoring Applications


- Software designed for creation of multimedia projects.
- Applications are used to:
 - Assemble media elements
 - Synchronize content
 - Design user interface
 - Provide user interactivity



Authoring Metaphors


- Authoring applications are grouped around three metaphors:
 - Card
 - Icon
 - Timeline
- Metaphors help orient developer to how the software organizes the media, sequences events, and presents final project.






Card Metaphor


- Media is organized in sequential order on a stack of cards or slides. (PowerPoint)
 - Appropriate for static media that is normally experienced in sequence.
- Cards have two layers:
 - **Background** layer contains shared elements.
 - **Foreground** layer contains content specific to that card or slide.



Benefits of Card Metaphor


- Benefits of card layers.
 - Background content is created once, which saves development time.
 - Common background layer provides consistent design.
 - File sizes are minimized by sharing background elements.





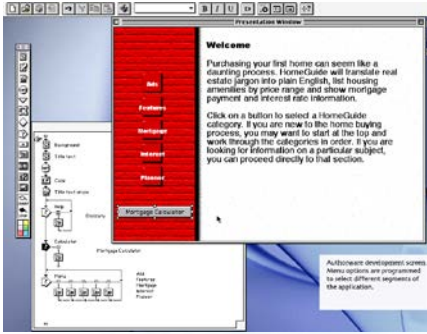
Icon Metaphor


- (Adobe Authorware)
- Icons define media and forms of interactivity.
- Icons are placed on a **flowline** to create the application structure.
 - Each icon has a dialog box with properties and parameters identified by the developer.
 - Flowlines let developers visualize and adjust the structure of the application.
- Branching routines add controls for user interaction.



Icon Metaphor



- **Flowline** is a graphical representation of the relationships between components of the application.





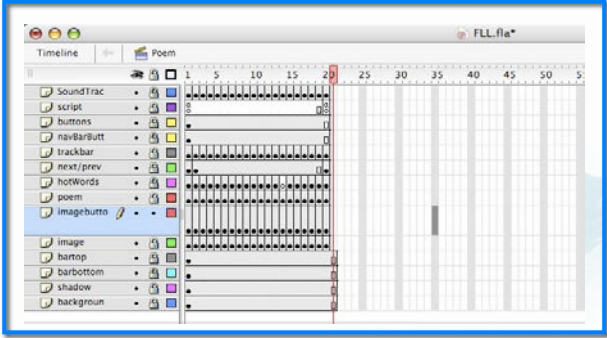
Timeline Metaphor

- Organizes media and interactivity as sequence of frames.
 - Each frame can have multiple layers.
 - Layers define the stacking order of the content to be displayed.
- Appropriate for dynamic media as the media can be synchronized precisely over time.

Timeline Metaphor

- Popular timeline-based applications include Director and Flash.
- Best used when animation or video is central to the application.




The screenshot shows a software interface with a timeline. The timeline is a horizontal axis with a grid, ranging from 0 to 55 seconds. On the left, there is a list of layers: SoundTrac, script, buttons, navBarButt, trackbar, next/prev, hotWords, poem, imagebutto, image, bartop, barbottom, shadow, and backgroun. Each layer has a corresponding track on the timeline with various colored bars and markers indicating the duration and synchronization of different elements.

A slide titled "Application Design" with a blue header and a background image of a laptop and a headset. The title is in white text.

- Create Flowchart and Storyboards
- Authoring software can establish the order of the content on playback.
- Basic navigation structures include:
 - Linear or sequential
 - Hierarchical
 - Networked
 - Conditional.



A diagram illustrating three navigation structures: Linear Navigation (a vertical stack of four boxes with downward arrows), Networked Navigation (a grid of six boxes with arrows connecting them in a non-linear path), and Hierarchical Navigation (a tree structure with a root box at the top and three levels of sub-boxes below it).

```
graph TD; subgraph Linear_Navigation; L1[ ] --> L2[ ]; L2 --> L3[ ]; L3 --> L4[ ]; end; subgraph Networked_Navigation; N1[ ] --> N2[ ]; N1 --> N3[ ]; N2 --> N4[ ]; N3 --> N4[ ]; N4 --> N5[ ]; end; subgraph Hierarchical_Navigation; H1[ ] --> H2[ ]; H1 --> H3[ ]; H2 --> H4[ ]; H2 --> H5[ ]; H3 --> H6[ ]; H3 --> H7[ ]; end;
```



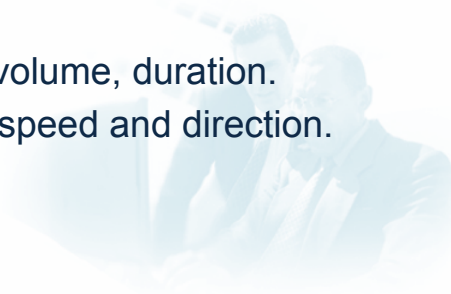
Importing Content


- Media is generally created in media-specific applications and imported into the authoring environment.
 - File formats for imported media are important.
 - Conversion utilities within the application are useful.



Create and Edit Content


- All authoring applications include some tools for creating and editing media content. For example:
 - Text adjustments to font size and color.
 - Paint tools to add shapes and edit image features.
 - Sound adjustment on volume, duration.
 - Animation changes to speed and direction.






Integrating, Synchronizing and Playback

- Techniques for integration are based on the metaphor (card, icon, timeline).
- Sounds, animations and transitions must be synchronized to present a unified flow of information.
- Playback of the content is often dependent on hardware factors. Timing controls can be established to ensure correct playback.




Programming

- Provides more flexibility and control.
 - For projects with extensive interactivity, custom features.
- Two programming methods.
 - **Script**: series of commands specifying properties or behavior of an element in the project.
 - Commands are interpreted as the project is executed.
 - **Icon**: dialog boxes allow the developer to specify parameters for icon's use.
 - Does not require programming knowledge but does limit commands to icon parameters.



Database Support

- Some projects may require access to a collection of related files to store and retrieve user input.
 - Tutorials have databases of related facts to test comprehension.
 - User stores answers for future reference and scoring.
- Authorware and Director applications offer an interface to a database.




Preview, Test, Debug

- Projects are created in the development mode.
- Necessary to **preview** the project as it will appear in the final product and **test** the components of the screen displays.
- Authoring applications often have a preview mode to test the assembled project during development.
- **Debugger** tools can identify errors in program code.




Project Delivery

- Projects are published so they play outside the authoring environment.
- Approaches to publishing:
 - Project requires a separate **player** program to present the multimedia content.
 - QuickTime, Flash, and MediaPlayer programs are free player downloads.
 - Project embeds the player in the multimedia project.
 - Larger files, but project is a stand-alone application.
 - Project plays at web browser



Choosing and Authoring Application

- No single authoring tool is suitable for all projects. To select the right application:
 - Consider the subject (static or dynamic media).
 - Consider the media (source file formats compatible).
 - Consider delivery (where used, means of distribution).
 - Consider maintenance (expertise needed to revise content, frequent update cycles).



Discussions

- Diskusikan keuntungan dan kerugian menggunakan **authoring tools** dalam membuat aplikasi pembelajaran.
- Diskusikan mengapa **multimedia** perlu diakomodasi dalam aplikasi pembelajaran.
- Diskusi diposting dalam forum diskusi Besmart. Tiap mahasiswa menanggapi 2 hal tsb.

