What is an *Applet?*

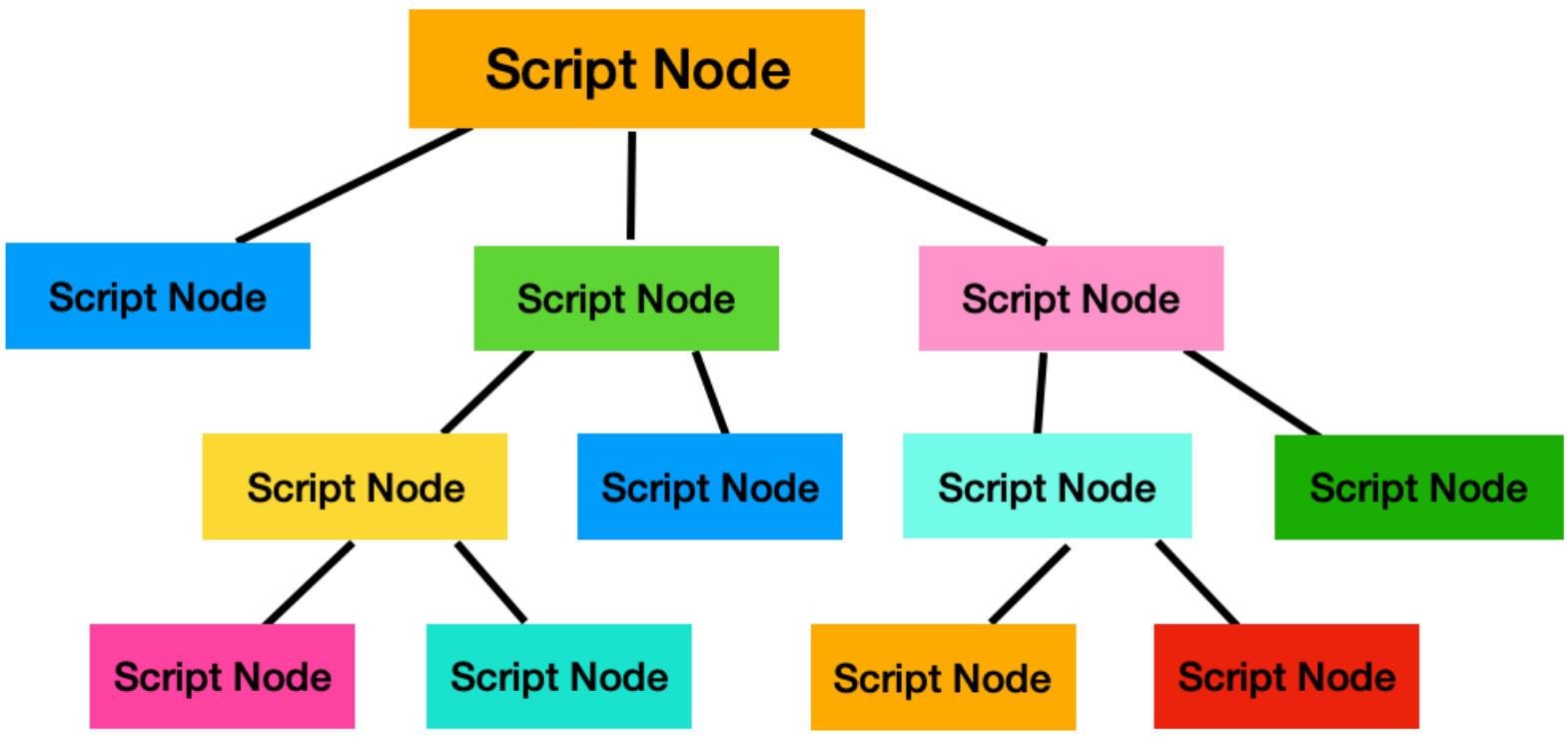
An *applet* is a set of coded *instructions* that are followed in order to create a *user experience* on a device. It is like an "*app*", but with two important differences:

1. An *app* is a compiled program. Its instructions must all be *compiled*, or *translated* into a machine-readable binary format before they can be run on the device. An *applet's* instructions are human-readable, and do not require compilation. Performing an *applet* requires a *player app* (called "WayneScriptPlayer"). The *WayneScript player* performs by reading and *interpreting* each instruction, then *rendering* each instruction in turn.

2. An *app* must be downloaded and installed on a device before it can be used. For Apple iPhones and iPads, an *app* must also be submitted to the corporation's App Store for approval. It must then be downloaded from the App Store to be installed on the device. To run an *applet*, only the *player app* must be downloaded and installed. Thereafter, any correctly-coded *applet* may be instantly run by the *player*.

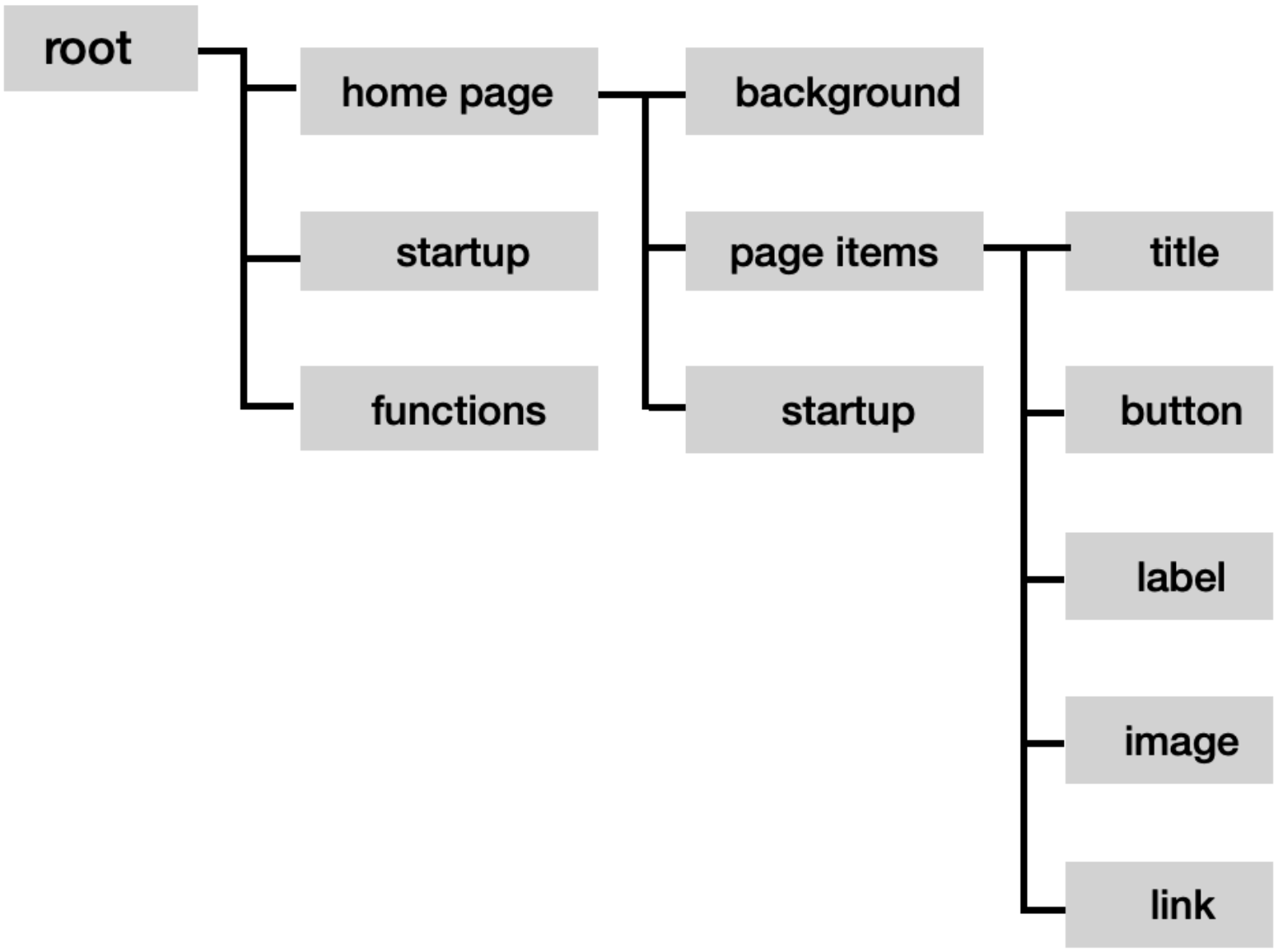
Structure of a *Script*

A *Script* is the group of instructions that define an applet. The instructions in a *script* do not form a single linear list, however. Instead, the *applet script* forms a logical hierarchy that resembles an organizational chart with an inverted tree structure. We appropriately call the units, or nodes, of the diagram *script nodes*. A complete *applet* then appears as a network of these *script nodes*.



It is clear from the diagram that an individual *script node* may have multiple descendants, or "*children*", but no more than one ancestor, or "*parent*". If a *script node* has no *children* beneath it, it is considered a "leaf", or *scalar*. On the other hand, if a node has one or more *children*, it can be called a *collection*. Consequently, every *script node* is either a *scalar* or a *collection*.

A *scalar* script node may be one of several types. The most common scalar types include *numeric* (integer or float), *string* (text), and *boolean* (true or false), along with a few other specialties such as point, rectangle, and color. A *collection* script node is always either an *array* or *dictionary*. Both of these *collection* types contain a list of script nodes that comprise its *children*. The difference between an *array* and *dictionary* is that an *array* is a numbered list that has a strict order. One identifies a member of an *array* by its numeric position in the list. A *dictionary*, on the other hand, has no specified order, but identifies its members with a unique name, or *key*. The structure and composition of *script nodes* will become more clear as you begin to see examples of them in actual scripts.
Every *WayneScript applet* will contain a single top-most *script node* which may be called its *root*. The *children* of the *root node* will typically include a *home page* and optional lists of *startup* instructions and user-defined *classes* and *functions*. The nature and utility of any *script node* defines the range and requirements of its own *child nodes* (if any). The diagram below shows a partial *script node* hierarchy for a simple, typical, exemplary *applet*:



Structure of an *Applet*

Every *applet* contains a single *root node* at the top of his hierarchy. The *root node's children* may contain optional user-defined *classes* and *functions*, which will be explained separately. The most essential ingredient of every *applet* (and its *root node*) is a hierarchy of *pages*. A *page* is the instructional unit that defines the visible device screen at any instant. Most typically, an *applet* will contain a single *home page* that will serve as the very first *page* to be displayed when an *applet* is launched. The *home page* then may contain *sub-pages* that can be navigated to from the *home page*. The *sub-pages* in turn may also contain their own *sub-pages*, forming a *page hierarchy*. A user will generally navigate between *pages* by tapping buttons, links, and tabs.

An *applet page* possesses several obvious attributes. One of these is its *background*, which will generally be a color or image. The most central and obvious feature of a *page* is the composition of its member *items*. (A *page item* is not to be confused with a s*ub-page*, which is an entire page that appears separately as the target of a navigation.) Each *page item* is a visible object that appears on top of the *page* at an arbitrary *position* and *size*. Typical *page items* include text fields, buttons, images, and a variety of familiar controls and widgets such as sliders, pickers, and selectors. A *page item* may also itself be a container, or *view*, that may hold other *items* within its own boundary. The *sizes* and *positions* of every *page item* can change dynamically. *Items* can be programmatically resized and shifted: either by animation or user actions such as dragging with the finger.

User-Programmable *Actions*

An *applet* would be severely limited, or nearly useless without being able to perform programmable tasks and *actions*. Such *actions* are the essence of computer programs. While *WayneScript* is not itself a programming language, it necessarily enjoys the capability of defining and enabling a large variety of tasks and *actions*. An *action* in *WayneScript* may constitute a single *script node*; but it will generally be an *array* of *script nodes* that are themselves *actions*. An *action* is thereby a hierarchy of *script nodes* that collectively perform a task. Some of the common *actions* include:

* Arithmetic and logical operations
* Assignment of values to variables
* Navigation between pages

* Creating pages and page items
* Controlling the size and position of items
* Setting and inspecting properties of pages and items
* Performing system or user-defined functions
* Conditional statements (if-then-else)
* Program loops
* Setting timers
* Responding to timing and other external events
* Responding to user interactions (button taps, dragging)
* Downloading files and data from web servers
* Displaying and manipulating images
* Playing sounds
* Drawing on the display
* Linking to other external applets

Script *actions* and tasks may be triggered and initiated in several ways. The *root script node* that defines an *applet* may optionally contain a “Startup” routine that will appear as an *array* of *action* nodes. An applet *page* or *sub-page* may similarly contain its own “Startup” *script*. Script *actions* will also be triggered by user interactions such as button taps, finger drags, and manipulation of screen controls. They may also be triggered by timer “ticks” and external events such as the completion of a download task.

Functions

Applets, like all computer programs, greatly vary in complexity. To be understandable, manageable, and maintainable, a complex computer program needs to be *modular*. Programming environments universally allow and facilitate creation and maintenance of individual autonomous *functions* that encapsulate and focus on a specific targeted goal. A computer algorithm will usually invoke many autonomous *functions* to complete its task. The *functions* will themselves invoke other more specific functions in their own control flow. The skillful exploitation of *functions* and *modularity* is a staple of good software design. *WayneScript* is obligingly no exception to this rule, as it allows and encourages liberal use of user-defined *functions*.

Classes and Objects

Object-oriented programming is a central capability of modern software environments. It is an extremely powerful discipline that is universally exploited in serious software. The theory, principles, and application of *object orientation* are not explained in this document, but they deserve the study, attention, and mastery of all programmers. The *WayneScript* platform provides and supports the definition and application of user-defined *Classes*. These *classes* enable the programmer to define and implement useful specialized objects that characterize a well-defined set of particular *properties* and *functions*. A *Class* serves as the definition and template that used to generate its *objects*. The *objects* then become program entities that may be assigned to *variables*, perform *actions,* and act as the players in the dynamics of a program.

Summary

A *WayneScript applet* is a set of programmable instructions that can be performed on a computational device. The *applet* comprises a hierarchy of instructional units, called *Script Nodes*, that form a logical inverted-tree hierarchy. An *applet* is composed of entities such as *functions*, *classes*, *pages*, *page items*, and *actions* that characterize its appearance, utility, and behavior. These entities are correspondingly represented by the *script nodes* that constitute the *applet*.