DraftCharterForWebComponentCommons

Draft Charter For Web Component Commons

About This Page

This is a space for easy development of the charter. Since the web components commons is intended to function in the same way that Jakarta Commons does, the initial version will be the current Jakarta Commons charter.

Please do not edit comments into this text: please use the CharterForWebCommonsRequestForComments or post to General At Jakarta.

Draft Charter

(0) rationale

Apache-Java and Jakarta originally hosted product-based subprojects, consisting of one major deliverable. The Java language however is package-based, and most of these products have many useful utilities. Some products are beginning to break these out so that they can be used independently. A Jakarta subproject to solely create and maintain independent packages is proposed to accelerate and guide this process.

(1) scope of the subproject

The subproject shall create and maintain packages written in the Java language, intended for use in server-related development, and designed to be used independently of any larger product or framework. Each package will be managed in the same manner as a larger Jakarta product. To further this goal, the subproject shall also host a workplace for Jakarta committers.

(1.1) the sandbox

The subproject will host a SVN repository available to all Apache committers as a workplace for new packages or other projects.

(2) identify the initial set of committers

PLEASE LEAVE EMPTY

Draft Guidelines

Note :

- is, has, will, shall, must - required.
- may, should, are encouraged - optional but recommended.

1. The primary unit of reuse and release is the package.
    a. The package library is not a framework but a collection of components designed to be used independently.
    b. Each package must have a clearly defined purpose, scope, and API – Do one thing well, and keep your contracts.
    c. Each package is treated as a product in its own right.
        i. Each package has its own status file, release schedule, version number, QA tests, documentation, **DELETED** *mailing list,* bug category, and individual JAR. 2. Each package must clearly specify any external dependencies, including any other Commons packages, and the earliest JDK version required.
            1. External dependencies on optional and third-party codebases should be minimized. 2. All necessary dependencies must be recorded in the MANIFEST.MF file of the package JAR, in the manner recommended in the JDK 1.3 documentation describing 'system extensions'
               3. Each package must maintain a list of its active committers in its status file.
    d. The packages should use a standard scheme for versioning, QA tests, and directory layouts, and a common format for documentation and Ant build files.
    e. The packages should fit within a unified package hierarchy.
    f. In general, packages should provide an interface and one or more implementations of that interface, or implement another interface already in use.
        - The packages should have boring functional names. Implementations may choose more 'exciting' designations.
    g. **DELETED** *Packages are encouraged to either use JavaBeans as core objects, a JavaBean-style API, or to provide an optional JavaBean wrapper.*
    h. External configuration files are discouraged, but if required, XML format files are preferred for configuration options.
    i. Each package will be hosted on its own page on the subproject Web site, and will also be indexed in a master directory.
    j. **DELETED** *The subproject will also host a top-level 'general' mailing list in addition to any lists for specific packages.* **ADDED** The subproject will provide two mailing lists: one for users and the other for developers. Posters should prefix the subject with the name of the component to allow easy filtering.
    k. **DELETED** *The subproject will also provide a single JAR of all stable package releases. It may also provide a second JAR with a subset of only JDK 1.1 compatible releases. A gump of nightly builds will also be provided.*
    l. Volunteers become committers to this subproject in the same way they are entered to any Jakarta subproject. Being a committer in another Jakarta subproject is not a prerequisite.
    m. Each committer has karma to all the packages, but committers are required to add their name to a package's status file before their first commit to that package.
    n. **DELETED** *New packages may be proposed to the Jakarta Commons mailing list. To be accepted, a package proposal must receive majority approval of the subproject committers. Proposals are to identify the rationale for the package, its scope, its interaction with other packages and products, the Commons resources, if any, to be created, the initial source from which the package is to be created, the coding conventions used for the package (if different from the Sun coding conventions), and the initial set of committers.*

** As stated in the Jakarta guidelines, an action requiring majority approval must receive at least 3 binding +1 votes and more +1 votes than -1 votes.*

o. It is expected that the scope of packages may sometimes overlap.

p. **DELETED** *Anyone may propose a new package to the Commons, and list themselves as the initial committers for the package. The vote on the proposal is then also a vote to enter new committers to the subproject as needed.*

q. **DELETED** *A SVN repository will be available to all Apache committers as a workplace for new packages or other projects. Before release to the public, code or documentation developed here must be sponsored by Jakarta subproject. The sponsoring subproject(s) will distribute the code or documentation along with the rest of their codebase.*

r. Each Commons component should use an internally consistent and documented coding style. When the source code for a component originates in a pre-existing code base outside of Commons, the coding style of that code base may be retained at the discretion of the initial committers. If a component does not specify its coding style, the Sun Coding Convention guidelines are assumed.

s. The subproject catalog will also list packages and resources available to the public related to other Jakarta subprojects and ASF projects.

t. As a Jakarta subproject, the Commons adopts all other guidelines and procedures of Jakarta and the Apache Software Foundation, as they may be amended from time to time.

u. Any member of the community may propose a new package. To be accepted, a package proposal must receive majority approval of the subproject committers and at least one committer must volunteer to serve as an initial package team member. Proposals should identify the rationale for the package, its scope, its interaction with other packages and products, the <insert-subproject-name> resources, if any, to be created, the initial source from which the package is to be created, and the sponsoring committers.

- As stated in the Jakarta guidelines, an action requiring majority approval must receive at least 3 binding +1 votes and more +1 votes than -1 votes.

v. The subproject will maintain an svn repository, referred to as the *sandbox,* as a workplace for new packages. Once approved, new packages must all begin in the sandbox. Any apache committer may contribute code directly to the sandbox and this code may form the initial source for new packages. Code from existing apache projects can, with the support of the contributing projects, also be imported directly into the sandbox. Finally, patches contributed incrementally by community members may be committed to the sandox by a subproject committer. If the initial source for a new package includes code from outside of apache, the new package must be brought into apache via the apache incubator.

w. A majority vote among subproject commiters is required to "graduate" a package from the "sandbox" to become a proper package. Only proper packages may make releases. Packages remaining in the sandbox for more than six months will require a majority vote to prevent archiving and removal from the subproject web site and any other public locations (e.g. nightly or continuous integration builds). Proper packages may not release code with dependencies on sandbox packages.

Up to CreatingCommonsForWebComponents