Properties Enhancement

Log4j 2 2.x uses the PropertiesUtil class in log4j-api to access properties. It was enhanced in Log4j 2.10.0 to "normalize" property names. While this helped, with the growth in the number of properties Log4j uses managing them is getting harder.

We also desire to be able to have components enabled or disabled by default but to provide settings to override the default. With Log4j 2 3.x being much more modularized enablement of many things can be controlled simply by the presence of the jar that contains the relevant functionality. However, it is sometimes the case that we will want individual plugins to require specific enablement to be activated. The structure (or lack of) the current set of properties makes this difficult.

This proposal is to require all properties to follow a more well defined structure. The proposal for the structure of a Log4j 2 property is log4j2.{LoggerContextName}.{ComponentName}.{property}. To keep the documentation from becoming a mess the current properties in 2.x would NOT be supported in 3.0.

The LoggerContext has always been allowed to have a name. it currently is internally generated and used as the unique key to locate it based on its associated ClassLoader. This is unfortunate as there are cases where users would like to specify properties that apply only to a specific LoggerContext. This proposal would create a new private LoggerContext field named contextKey that is used in place of the contextName as the key in the LoggerContext Map. The various API layers would be modified to accept a contextName field when he LoggerContext is created. The context name will be immutable.

This will allow a configuration like the one shown below. log4j2 at the top level identifies the file as containing log4j properties. The element at the next level down is the name of the LoggerContext the enclosed properties apply to. A "*" as the LoggerContext name indicates the settings are the default values. Within that properties are grouped by the components they apply to. Note that the term "component" is used loosely here.

Note that JsonTemplateLayout includes a JsonReader. If that is moved into log4j-core we should be able to support JSON configuration and properties without any additional dependencies. In addition, our Spring Boot support will allow the user to specify the JSON below in the application's bootstrap.yml file as an alternative to log4j2.component.properties and log4j2.component.yml.

```
{ "log4j2": {
    "My-App": {
      "JNDI": {
        "enableJMS": "true"
      }
      "Script": {
        "enableLanguages": [
              "Groovy",
          "JavaScript"
                ]
      }
      "Configuration": {
        "mergeStrategy": "com.acme.log4j.CustomMergeStrategy",
        "location": "classpath:log4j2-My-App.xml",
        "statusLoggerLevel": "debug"
      }
   },       "*":  {
            "StatusLogger": {
                "defaultStatusLevel": "Info"
            }
       }}
```

Of course, the equivalent would also be available as properties, although the property names will change from their current values

```
log4j2.My-App.Jms.enableJndi=true
log4j2.My-App.Script.enableLanguages=Groovy,Javascript
log4j2.My-App.Configuration.mergeStrategy=com.acme.log4j.CustomMergeStrategy
log4j2.My-App.Configuration.location=classpath:log4j2-My-App.xml
log4j2.My-App.Configuration.statusLoggerLevel=debug
log4j2.*.StatusLogger.defaultStatusLevel=Info
```

Using this format allows restrictions to be placed on plugins:

```
@RequireProperty(component = "Lookup", property = "enableJndi", value = "true")
@Plugin(name = "jndi", category = StrLookup.CATEGORY)
public class JndiLookup extends AbstractLookup {}
```

This would prevent the Plugin from being loaded unless the requirement is met.

In the case of Scripting we would want the ScriptManager to be disabled unless one or more scripting language are enabled. In the case of the ScriptManager it is loaded by the ScriptManagerFactory in Log4j 2 3.x, which is loaded via ServiceLoader. The ScriptManager then locates all the ScriptEngines it can find. Each engine identifies the scripting languages it supports. If a matching language isn't specified in the properties than that engine would be discarded.

Log4j 2 3.x will also come with a file named log4j2.default.component.yml that declares the default values for all properties except for log4j2.debug, log4j2.contextSelector, log4j2.ignoreTCL, and log4j2.forceTCLOnly whose values apply to all usages of Log4j in the JVM. The file below will be included in Log4j 2 to contain the default values.

```
{
        "log4j2": {
                "*": {
                        "AsyncLogger": {
                                "exceptionHandler": "",
                                "ringBufferSize": "caclculate",
                                "waitStrategy": "Timeout",
                                "timeout": 10,
                                "sleepTimeNS": 100,
                                "retries": 200,
                                "synchronizeEnqueWhenFull": true,
                                "threadNameStrategy": "CACHED",
                                "queueFullPolicy": "",
                                "discardThreshold": "INFO",
                                "formatMsg": false
                        },
                        "Configuration": {
                                "location": "",
                                "mergeStrategy": "org.apache.logging.log4j.core.config.composite.
DefaultMergeStrategy",
                                "configurationFactory": "org.apache.logging.log4j.core.config.
ConfigurationFactory",
                                "clock": "org.apache.logging.log4j.core.util.SystemClock",
                                "level": "ERROR",
                "allowedProtocols": ["HTTPS"]
                        },
                        "GC": {
                                "enableThreadLocals": true,
                                "enableDirectEncoders": true,
                                "intialReusableMsgSize": 128,
                                "maxReusableMsgSize": 518,
                                "layoutStringBuilderMaxSize": 2048,
                                "unboxRingBufferSize": 32
                        },
                        "Jansi": {
                                "enabled": "true"
                        },
                        "JMX": {
                                "enabled": "false",
                                "notifyAsync": "calculate"
                        },
                        "JNDI": {
                                "enableContextSelector": "false",
                                "enableJdbc": "false",
                                "enableJMS": "false",
                                "enableLookup": "false"
                        },
                        "JUL": {
                                "loggerAdapter": "org.apache.logging.log4j.jul.ApiLoggerAdapter"
                        },
                        "LoggerContext": {
                                "logEventFactory": "org.apache.logging.log4j.core.impl.DefaultLogEventFactory",
                                "loggerContextFactory": "org.apache.logging.log4j.simple.
SimpleLoggerContextFactory",
                                "shutdownHookEnabled": "true",
                                "shutdownCallbackRegistry": "org.apache.logging.log4j.core.util.
DefaultShutdownCallbackRegistry",
                                "stackTraceOnStart": false
                        },
                        "Message": {
                                "messageFactory": "org.apche.logging.log4j.message.ParameterizedMessagefactory",
```

```
                                "flowMessageFactory": "org.apache.logging.log4j.message.
DefaultFlowMessageFactory",
                                "jsonFormatterMaxDepth": 8
                        },
                        "Script": {
                                "enableLanguages": []
                        },
                        "SimpleLogger": {
                                "showContextMap": false,
                                "showLogName": false,
                                "showShortLogName": true,
                                "showDateTime": false,
                                "dateTimeFormat": "yyyy/MM/dd HH:mm:ss:SSS zzz",
                                "logFile": "systemerr",
                                "logLevel": "ERROR",
                                "{loggerName}.level": "",
                                "statusLoggerLevel": "ERROR"
                        },
                        "StatusLogger": {
                                "defaultStatusLevel": "ERROR",
                                "statusLoggerLevel": "WARN",
                                "entries": 200,
                                "dateFormat": ""
                        },
                        "ThreadContext": {
                                "enabled": "true",
                                "enableMap": "true",
                                "enableStack": "false",
                                "inheritable": "false",
                                "garbageFree": "false",
                                "initialCapacity": 16,
                                "contextDataInjector": "org.apache.logging.log4j.core.ContextDataInjector"
                        },
                        "TransportSecurity": {
                                "trustStoreLocation": "",
                                "trustStorePassword": "",
                                "trustStorePasswordFile": "",
                                "trustStorePasswordEnvironmentVariable": "",
                                "trustStoreType": "trustStoreKeyManagerFactoryAlgorithm",
                                "keyStoreLocation": "",
                                "keyStorePassword": "",
                                "keyStorePasswordFile": "",
                                "keytorePasswordEnvironmentVariable": "",
                                "keyStoreType": "",
                                "keyStoreKeyManagerFactoryAlgorithm": ""
                        },
                        "UUID": {
                                "sequence": ""
                        },
                        "Web": {
                                "enableWebApp": "calculate"
                        }
                }
        }
}
```