Chapter 8 - SHRINE's Configuration File

SHRINE gets its configuration from a configuration file named **shrine.conf**. If you are installing a downstream (non-hub) node, copy the shrine-setup/qep-and-adapter-shrine.conf file to tomcat's lib directory. If you are creating a hub use hub-and-qep-shrine.conf instead.

```
cp shrine-setup/qep-and-adapter-shrine.conf /opt/shrine/tomcat/lib/shrine.conf
```

In this guide, we will refer to this file often and will go more in detail on configuring this file in the later chapters). Here is an example shrine.conf file (it might be a good idea to copy the entire contents of this file and then replace your own specific values in):

```
shrineHubBaseUrl = "https://shrine-hub.faraway.com:6443" //The shrine hub's URL as observed from this tomcat server
i2b2BaseUrl = "http://i2b2.example.com:9090"         //The local i2b2's URL as observed from this tomcat server
i2b2Domain = "exampleDomain"
i2b2ShrineProjectName = "SHRINE"

shrine {

  nodeKey = "testNode" //node key to get information from the hub about this node. Your hub administrator will tell you this value
  shrineDatabaseType = "mysql" //can be mysql, sqlserver, oracle

  messagequeue {
    blockingq {
      serverUrl = ${shrineHubBaseUrl}/shrine-api/mom
    }
  }

  webclient {
   domain = ${i2b2Domain}
   name  = ${i2b2ShrineProjectName}
   siteAdminEmail = "email@example.com"
   usernameLabel = "User Name"
   passwordLabel = "User Password"
   defaultNumberOfOntologyChildren = 10000  // the number of children to show when an ontology folder is expanded.
   queryFlaggingInstructions = "Enter instructions for flagging queries here"
   flaggingPlaceholderText = "Enter placeholder text for the query flagging text input field"
   flaggingIconInstructions = "Enter text for when user mouses over the flagging information icon in the header of the Query History here"
  }

  pmEndpoint {
    url = ${i2b2BaseUrl}/i2b2/services/PMService/getServices
  }

  ontEndpoint {
    url = ${i2b2BaseUrl}/i2b2/services/OntologyService
  }

  hiveCredentials {
    domain = ${i2b2Domain}
    username = "demo"
    password = "demouser"
    crcProjectId = "Demo"
    ontProjectId = ${i2b2ShrineProjectName}
  }

  breakdownResultOutputTypes {
    PATIENT_AGE_COUNT_XML {
      description = "Age patient breakdown"
    }
    PATIENT_RACE_COUNT_XML {
      description = "Race patient breakdown"
    }
    PATIENT_VITALSTATUS_COUNT_XML {
      description = "Vital Status patient breakdown"
    }
    PATIENT_GENDER_COUNT_XML {
      description = "Gender patient breakdown"
    }
  } //end breakdown section

  hub {
    client {
      serverUrl = ${shrineHubBaseUrl}
    }
  }

  queryEntryPoint {
    broadcasterServiceEndpoint {
      url = ${shrineHubBaseUrl}/shrine/rest/broadcaster/broadcast
    }
  }

  adapter {
    crcEndpoint {
      url = ${i2b2BaseUrl}/i2b2/services/QueryToolService/
    }

    adapterMappingsFileName = "AdapterMappings.csv"
    crcRunQueryTimeLimit = "30 seconds" // in seconds, use quotes. default 30 seconds
  } //end adapter section

  keystore {
    file = "/opt/shrine/shrine.keystore"
    password = "changeit"
    privateKeyAlias = "shrine-node"
    keyStoreType = "JKS"
    caCertAliases = ["shrine-ca"]
  } //end keystore section

  steward {
    createTopicsMode = Approved //the default is Pending - the most secure - but most sites use Approved

    emailDataSteward {
      sendAuditEmails = false  //false to turn off the whole works of emailing the data steward
          //interval = "1 day" //Audit researchers daily
      //timeAfterMidnight = "6 hours" //Audit researchers at 6 am. If the interval is less than 1 day, then
this delay is ignored.
      //maxQueryCountBetweenAudits = 30 //If a researcher runs more than this many queries since the last audit
audit her
      //minTimeBetweenAudits = "30 days" //If a researcher runs at least one query, audit those queries if this
much time has passed

      //You must provide the email address of the shrine node system admin, to handle bounces and invalid
addresses
      //from = "shrine-admin@example.com"
      //You must provide the email address of the data steward
      //to = "shrine-steward@example.com"

      //subject = "Audit SHRINE researchers"
      //The baseUrl for the data steward to be substituted in to email text. Must be supplied if it is used in
the email text.
      //stewardBaseUrl = "https://example.com:6443/shrine-api/shrine-steward/"
      externalStewardBaseUrl = "https://example.com:6443/shrine-api/shrine-steward/"

      //Text to use for the email audit.
      //AUDIT_LINES will be replaced by a researcherLine for each researcher to audit.
      //STEWARD_BASE_URL will be replaced by the value in stewardBaseUrl if available.
      //emailBody = """Please audit the following users at STEWARD_BASE_URL at your earliest convenience:
AUDIT_LINES"""
      //note that this can be a multiline message

      //Text to use per researcher to audit.
      //FULLNAME, USERNAME, COUNT and LAST_AUDIT_DATE will be replaced with appropriate text.
      //researcherLine = "FULLNAME (USERNAME) has run COUNT queries since LAST_AUDIT_DATE."
    }
  } //end steward section

} //end shrine
```

nodeKey parameter

The nodeKey parameter will be used to identify your node from the Hub, so we advise that it should be a relatively unique identifier along with the network in which you are in, ie. HarvardProdNode or similar. If you have any questions, please contact the network administrator for more information. You'll also need to tell the admin your domain parameter, so that the domain will be associated with the nodeKey on the hub.