

Modul:Namespace detect/data

This is a data page for [Module:Namespace detect](#) and [Module:Category handler/shared](#). It is loaded by the main module using `mw.loadData`, which means it is only processed once per page rather than once per `#invoke`.

```
-----  
--                               Namespace detect data                               --  
-- This module holds data for [[Module:Namespace detect]] to be loaded per      --  
-- page, rather than per #invoke, for performance reasons.                       --  
-----  
  
local cfg = require('Module:Namespace detect/config')  
  
local function addKey(t, key, defaultKey)  
    if key ~= defaultKey then  
        t[#t + 1] = key  
    end  
end  
  
-- Get a table of parameters to query for each default parameter name.  
-- This allows wikis to customise parameter names in the cfg table while  
-- ensuring that default parameter names will always work. The cfg table  
-- values can be added as a string, or as an array of strings.  
  
local defaultKeys = {  
    'main',  
    'talk',  
    'other',  
    'subjectns',  
    'demospace',  
    'demopage'  
}  
  
local argKeys = {}  
for i, defaultKey in ipairs(defaultKeys) do  
    argKeys[defaultKey] = {defaultKey}  
end  
  
for defaultKey, t in pairs(argKeys) do  
    local cfgValue = cfg[defaultKey]  
    local cfgValueType = type(cfgValue)  
    if cfgValueType == 'string' then  
        addKey(t, cfgValue, defaultKey)  
    elseif cfgValueType == 'table' then  
        for i, key in ipairs(cfgValue) do  
            addKey(t, key, defaultKey)  
        end  
    end  
    cfg[defaultKey] = nil -- Free the cfg value as we don't need it any more.  
end  
  
local function getParamMappings()  
    --[[  
    -- Returns a table of how parameter names map to namespace names. The key  
    -- are the actual namespace names, in lower case, and the values are the  
    -- possible parameter names for that namespace, also in lower case. The  
    -- table entries are structured like this:  
    -- {
```



```
--    [''] = {'main'},
--    ['wikipedia'] = {'wikipedia', 'project', 'wp'},
--    ...
-- }
--]]
local mappings = {}
local mainNsName = mw.site.subjectNamespaces[0].name
mainNsName = mw.ustring.lower(mainNsName)
mappings[mainNsName] = mw.clone(argKeys.main)
mappings['talk'] = mw.clone(argKeys.talk)
for nsid, ns in pairs(mw.site.subjectNamespaces) do
    if nsid ~= 0 then -- Exclude main namespace.
        local nsname = mw.ustring.lower(ns.name)
        local canonicalName = mw.ustring.lower(ns.canonicalName)
        mappings[nsname] = {nsname}
        if canonicalName ~= nsname then
            table.insert(mappings[nsname], canonicalName)
        end
        for _, alias in ipairs(ns.aliases) do
            table.insert(mappings[nsname], mw.ustring.lower(alias))
        end
    end
end
return mappings
end

return {
    argKeys = argKeys,
    cfg = cfg,
    mappings = getParamMappings()
}
```