MediaWiki:Gadget-MassPatrol.js

Note: After publishing, you may have to bypass your browser's cache to see the changes.

- **Firefox / Safari:** Hold *Shift* while clicking *Reload*, or press either *Ctrl-F5* or *Ctrl-R* (*⌘-R* on a Mac)
- **Google Chrome:** Press *Ctrl-Shift-R* (*⌘-Shift-R* on a Mac)
- **Internet Explorer / Edge:** Hold *Ctrl* while clicking *Refresh*, or press *Ctrl-F5*
- **Opera:** Press *Ctrl-F5*.

```
/*
* Mass Patrol
* zdroj: https://fr.wikipedia.org/w/index.php?title=MediaWiki:Gadget-Smart_patrol.js&oldid=151659401
*/

if ( mw.config.get('wgNamespaceNumber') >= 0 ) {

    mw.loader.using( 'mediawiki.util', function () {
        $( function ( $ ) {

            var link = mw.util.addPortletLink( 'p-tb', '#', 'Mass patrol', 'Mass patrol' );
            $( link ).click( function ( e ) {
                e.preventDefault();
                mw.loader.using( [ 'mediawiki.api', 'mediawiki.notify', 'oojs-ui' ], function () {
                    askToLaunch();
                } );
            } );

            function askToLaunch() {
                new OO.ui.confirm( 'Do you want to mark all revisions as patrolled?' ).then( function ( response ) {
                    if ( response === true ) {
                        getAndProcessRevisions();
                    }
                } );
            }

            function getAndProcessRevisions( apicontinue ) {
                var params = {
                    'action': 'query',
                    'format': 'json',
                    'prop': 'revisions',
                    'titles': mw.config.get( 'wgPageName' ),
                    'formatversion': 2,
                    'rvprop': 'ids',
                    'rvlimit': 50
                };
                if ( apicontinue ) {
                    $.extend( params, apicontinue );
                }
                new mw.Api()
                    .get( params )
                    .then( function ( data ) {
                        var revisions = data.query.pages[ 0 ].revisions;
                        markRevisionsAsPatrolled( revisions )
                            .then( function ( shouldContinue ) {
                                if ( shouldContinue && data[ 'continue' ] ) {
                                    getAndProcessRevisions( data[ 'continue' ] );
                                } else {
                                    window.location.reload();
                                }
                            } )
                            .fail( function ( error ) {
                                mw.notify( 'Error: ' + error, { title: 'Mass patrol', type: 'error' } );
                            } );
                    } )
                    .fail( function ( error ) {
                        mw.notify( 'Error: ' + error, { title: 'Mass patrol', type: 'error' } );
                    } );
            }

            function markRevisionsAsPatrolled( revisions ) {
                var deferred = $.Deferred();
                markOneAsPatrolled( revisions, 0, deferred );
                return deferred.promise();
            }

            function markOneAsPatrolled( revisions, index, deferred ) {
                var revid = revisions[ index ].revid;
                new mw.Api()
                    .postWithToken( 'patrol', {
                        'action': 'patrol',
                        'revid': revid
                    } )
                    .then( function ( info ) {
                        console.log( 'Success: ' + revid );
                        if ( revisions[ index + 1 ] ) {
                            markOneAsPatrolled( revisions, index + 1, deferred );
                        } else {
                            deferred.resolve( true );
                        }
                    } )
                    .fail( function ( error ) {
                        if ( error === 'notpatrollable' ) {
                            console.log( "Error due to old revisions: " + revid );
                            deferred.resolve( false );
                        } else {
                            deferred.reject( error );
                        }
                    } );
            }

        } );
    } );
}
```