```
Linkedin = { config: { scrollDelay: 1000, actionDelay: 3000, nextPageDelay: 2000, // set to -1 for no limit maxRequests: 20, totalRequestsSent: 0, // set to false to skip adding note in invites addNote: false, note: "Hey {{name}}, I'm looking forward to connecting with you!", }, init: function (data, config) { console.info("INFO: script initialized on the page..."); console.debug( "DEBUG: scrolling to bottom in " + config.scrollDelay + " ms" ); setTimeout(() => this.scrollBottom(data, config), config.actionDelay); }, scrollBottom: function (data, config) { window.scrollTo({ top: document.body.scrollHeight, behavior: "smooth" }); console.debug("DEBUG: scrolling to top in " + config.scrollDelay + " ms"); setTimeout(() => this.scrollTop(data, config), config.scrollDelay); }, scrollTop: function (data, config) { window.scrollTo({ top: 0, behavior: "smooth" }); console.debug( "DEBUG: inspecting elements in " + config.scrollDelay + " ms" ); setTimeout(() => this.inspect(data, config), config.scrollDelay); }, inspect: function (data, config) { var totalRows = this.totalRows(); console.debug("DEBUG: total search results found on page are " + totalRows); if (totalRows >= 0) { this.compile(data, config); } else { console.warn("WARN: end of search results!"); this.complete(config); } }, compile: function (data, config) { var elements = document.querySelectorAll("button"); data.pageButtons = [...elements].filter(function (element) { return element.textContent.trim() === "Connect"; }); if (!data.pageButtons || data.pageButtons.length === 0) { console.warn("ERROR: no connect buttons found on page!"); console.info("INFO: moving to next page..."); setTimeout(() => { this.nextPage(config); }, config.nextPageDelay); } else { data.pageButtonTotal = data.pageButtons.length; console.info("INFO: " + data.pageButtonTotal + " connect buttons found"); data.pageButtonIndex = 0; var names = document.getElementsByClassName("entity-result__title-text"); names = [...names].filter(function (element) { return element.parentNode.parentNode.parentNode.parentNode.parentNode.parentNode.parentNode.parentNode.textContent.includes( "Connect\n" ); }); data.connectNames = [...names].map(function (element) { return element.innerText.split(" ")[0]; }); console.debug( "DEBUG: starting to send invites in " + config.actionDelay + " ms" ); setTimeout(() => { this.sendInvites(data, config); }, config.actionDelay); } }, sendInvites: function (data, config) { console.debug("remaining requests " + config.maxRequests); if (config.maxRequests == 0) { console.info("INFO: max requests reached for the script run!"); this.complete(config); } else { console.debug( "DEBUG: sending invite to " + (data.pageButtonIndex + 1) + " out of " + data.pageButtonTotal ); var button = data.pageButtons[data.pageButtonIndex]; button.click(); if (config.addNote && config.note) { console.debug( "DEBUG: clicking Add a note in popup, if present, in " + config.actionDelay + " ms" ); setTimeout(() => this.clickAddNote(data, config), config.actionDelay); } else { console.debug( "DEBUG: clicking done in popup, if present, in " + config.actionDelay + " ms" ); setTimeout(() => this.clickDone(data, config), config.actionDelay); } } }, clickAddNote: function (data, config) { var buttons = document.querySelectorAll("button"); var addNoteButton = Array.prototype.filter.call(buttons, function (el) { return el.textContent.trim() === "Add a note"; }); // adding note if required if (addNoteButton && addNoteButton[0]) { console.debug("DEBUG: clicking add a note button to paste note"); addNoteButton[0].click(); console.debug("DEBUG: pasting note in " + config.actionDelay); setTimeout(() => this.pasteNote(data, config), config.actionDelay); } else { console.debug( "DEBUG: add note button not found, clicking send on the popup in " + config.actionDelay ); setTimeout(() => this.clickDone(data, config), config.actionDelay); } }, pasteNote: function (data, config) { noteTextBox = document.getElementById("custom-message"); noteTextBox.value = config.note.replace( "{{name}}", data.connectNames[data.pageButtonIndex] ); noteTextBox.dispatchEvent( new Event("input", { bubbles: true, }) ); console.debug( "DEBUG: clicking send in popup, if present, in " + config.actionDelay + " ms" ); setTimeout(() => this.clickDone(data, config), config.actionDelay); }, clickDone: function (data, config) { var buttons = document.querySelectorAll("button"); var
```

```
doneButton = Array.prototype.filter.call(buttons, function (el) { return el.textContent.trim() === "Send"; }); // Click the first send button if (doneButton && doneButton[0]) { console.debug("DEBUG: clicking send button to close popup"); doneButton[0].click(); } else { console.debug( "DEBUG: send button not found, clicking close on the popup in " + config.actionDelay ); } setTimeout(() => this.clickClose(data, config), config.actionDelay); }, clickClose: function (data, config) { var closeButton = document.getElementsByClassName( "artdeco-modal__dismiss artdeco-button artdeco-button--circle artdeco-button--muted artdeco-button--2 artdeco-button--tertiary ember-view" ); if (closeButton && closeButton[0]) { closeButton[0].click(); } console.info( "INFO: invite sent to " + (data.pageButtonIndex + 1) + " out of " + data.pageButtonTotal ); config.maxRequests--; config.totalRequestsSent++; if (data.pageButtonIndex === data.pageButtonTotal - 1) { console.debug( "DEBUG: all connections for the page done, going to next page in " + config.actionDelay + " ms" ); setTimeout(() => this.nextPage(config), config.actionDelay); } else { data.pageButtonIndex++; console.debug( "DEBUG: sending next invite in " + config.actionDelay + " ms" ); setTimeout(() => this.sendInvites(data, config), config.actionDelay); } }, nextPage: function (config) { var pagerButton = document.getElementsByClassName( "artdeco-pagination__button--next" ); if ( !pagerButton || pagerButton.length === 0 || pagerButton[0].hasAttribute("disabled") ) { console.info("INFO: no next page button found!"); return this.complete(config); } console.info("INFO: Going to next page..."); pagerButton[0].click(); setTimeout(() => this.init({}, config), config.nextPageDelay); }, complete: function (config) { console.info( "INFO: script completed after sending " + config.totalRequestsSent + " connection requests" ); }, totalRows: function () { var search_results = document.getElementsByClassName("search-result"); if (search_results && search_results.length != 0) { return search_results.length; } else { return 0; } }, }; Linkedin.init({}, Linkedin.config);
```