Health Inspector – User Guide

Contents

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved

Overview

The Health Inspector application was designed to notify a defined set of users of the general "health" of your Salesforce instance's test classes and code coverage. The Health Inspector is an extremely lightweight application that sends an email notification, at a customizable interval, which documents:

- Any test class failures
- Confirmation of test classes running longer than a configurable second count
- The overall code coverage of the environment and whether it is below a user defined threshold
- Flow code coverage

Lastly it documents the long running Apex Classes & Methods which can be used not only as a reference but also as a baseline for future executions.

The Health Inspector app not only notifies a defined set of users but it can also generate a file which contains the results of the execution so you can monitor your results over time or refer back to any previous executions.

The application was designed to be administered with simple configuration updates, rather than via code, and can be completely managed via the Salesforce setup menu.

GearsDesign

Copyright© 2021 GearsDesign, Inc. All Rights Reserved

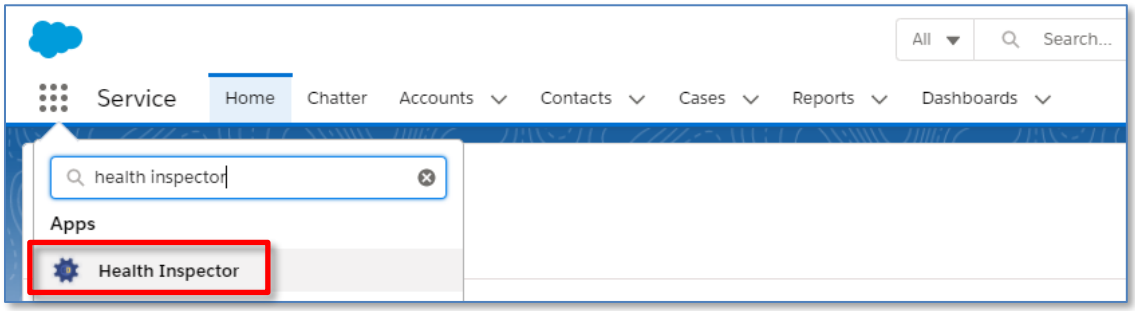
Health Inspector Installation

The Health Inspector package can be installed from the following AppExchange listing:
https://appexchange.salesforce.com/appxListingDetail?listingId=a0N3A00000FMkhsUAD

During the installation process, leave all selected defaults and install for Administrators only.

Health Inspector App

After installing Health Inspector, you will be able to select the Health Inspector app from the App Launcher:



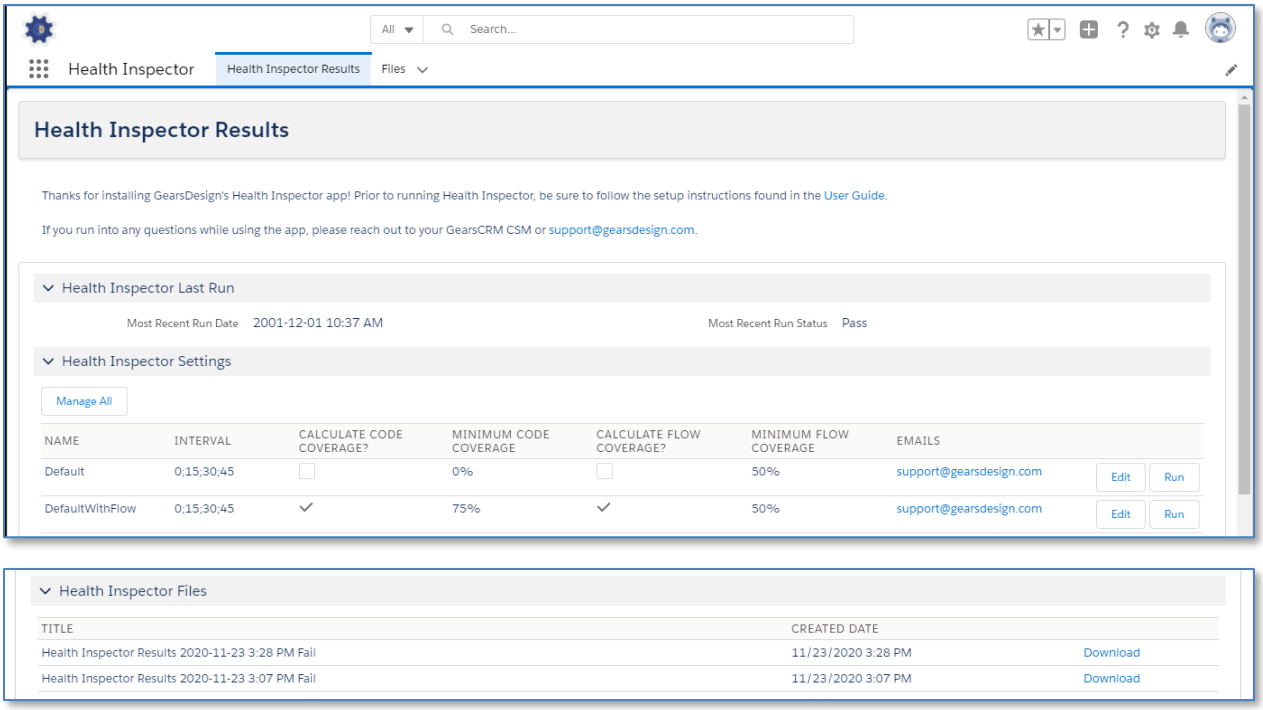
There are two tabs by default in this app – 'Health Inspector Results' & 'Files'.

Health Inspector Results

The Health Inspector Results tab is designed to be the "one stop shop" for managing the application. This screen provides information relevant to:

- The last execution of the Health Inspector app (pass or fail)
- The resulting status of the last execution
- Easy way to view all available configurations
- Quick links to manage all configurations or edit individually
- Ability to run a one-off execution of a configuration
- Quick access to any files generated from executions

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved



Files

This is the native Salesforce Files tab. If you have set Health Inspector to generate files on success/failure, you will be able to see those records on this tab.

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved

Health Inspector Configuration

Upon installation of the Health Inspector application, one Custom Metadata Type record is created:

- Test Class Audit Parameter (Test_Class_Audit_Parameters__mdt)

Additionally, there is one Custom Setting that can be configured to store details around the last execution of the application:

- Global Settings (Global_Settings__c)

Below is a summary of the Custom Metadata Type & Custom Setting and how they are used in conjunction with the application.

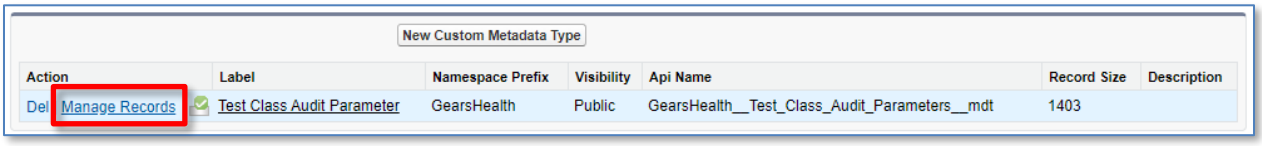
Test Class Audit Parameter (Test_Class_Audit_Parameters__mdt)

The 'Test Class Audit Parameter' Custom Metadata Type (CMDT) is used to drive the core functionality of the Health Inspector application. Within this CMDT, the following can be configured:

- Cadence of the app's execution
- Recipients of the email notifications
- Code coverage tolerance amount
- Contents of the success & failure email alerts
- Code coverage running user
- Generation of success or failure File creation

Upon installation of the application, two entries will be created: 'Default' & 'DefaultWithFlow'. The difference between these two are that the 'DefaultWithFlow' entry contains language in the template to support the Flow Code Coverage functionality. Whereas the 'Default' entry has language specific to standard code coverage – not including flow. These entries are fully customizable so they can be modified to your exact use case. There is also the option of creating a new entry to support your individual requirements.

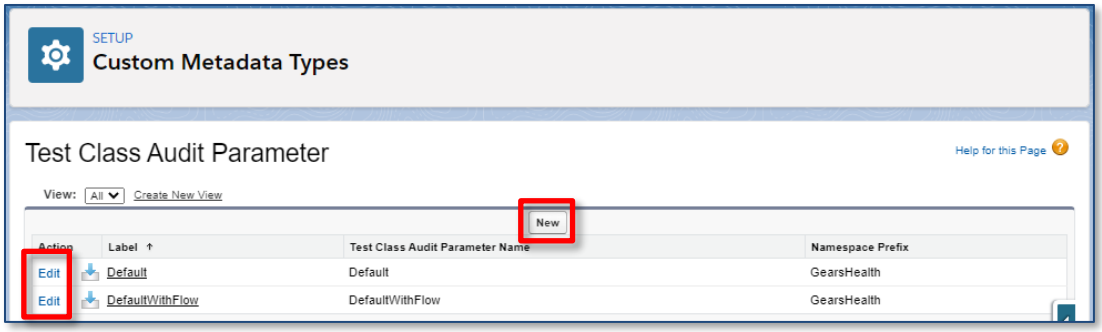
To create a new or update the existing CMDT entries, navigate to the Setup menu and type in 'Custom Metadata Types' in the search box and select that from the menu. In the resulting screen, find the 'Test Class Audit Parameter' custom metadata type and click the 'Manage Records' link next to it:



From the resulting screen, to create a new entry click the New button or to edit one of the existing, click the Edit link to the left of the 'Default' or 'DefaultWithFlow' entry. For most organizations, one of the Default entries should be used, as it contains the HTML for both the Health Inspector success and failure email templates. For users who are comfortable with HTML, you can feel free to create a new CMDT record and create your own email content.

GearsDesign

Copyright© 2021 GearsDesign, Inc. All Rights Reserved



In the next screen, the following field should be populated based on your requirements:

> ***** Important Note *****
>
> When upgrading Health Inspector, it will not add newly created fields to existing page layouts. If you don't see any of the fields listed below on your CMDT setup, it's likely because they just need to be added via the page layout.

- **Information Section**
  - ***Label - (Required)***
    - Enter an intuitive name for the Health Inspector entry
  - ***Test Class Audit Parameter Name – (Required)***
    - This will default based on the Label value that was provided
    - Note: you will need this value to schedule the Health Inspector in a subsequent step
  - ***Duration Tolerance (seconds)***
    - This should be the number of seconds the test should run – for example, if the test runs longer than X seconds, include it in the notification
    - For reference, 10 seconds is usually a baseline that can be used to determine if there is a long running class
  - ***Email Distribution***
    - Enter the email address or addresses (comma delimited) that should receive the Health Inspector email results
  - ***Ignore Errors***
    - Enter any errors that should be ignored by the Health Inspector logic. For example, NO_MAIL_PERMISSION, NO_MASS_EMAIL, etc.
  - ***Check Audit Increments – (Required)***
    - Enter the minute(s) on the hour that the Health Inspector should be run
    - For example, a value in this field of "1;15;30;45" during the 12:00 hour will execute at 12:01, 12:15, 12:30 & 12:45
  - ***Code Coverage Tolerance***

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved

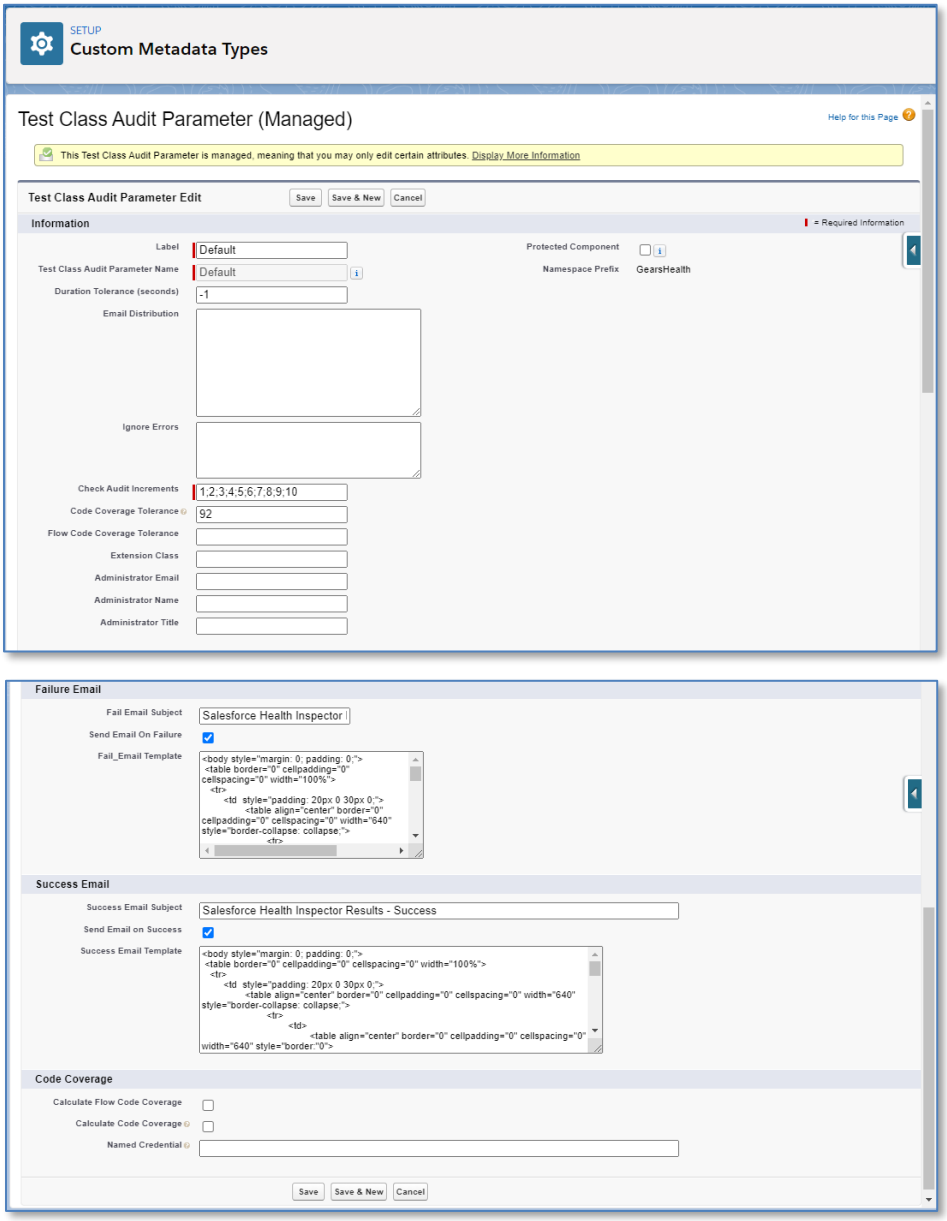
        - Enter a numeric value (up to 100) that indicates the code coverage percentage that should be used as a baseline to compare the org's overall test class coverage against.
        - For example, a value of "90" in this field would mean that if the overall code coverage of the org falls below 90%, this will also show as a notification in the Health Inspector email alert
    - ***Flow Code Coverage Tolerance***
        - Only applicable if 'Flow Code Coverage' is enabled
        - Enter a numeric value (up to 100) that indicates the flow code coverage percentage that should be used as a baseline to compare the org's overall flow code coverage against.
        - For example, a value of "90" in this field would mean that if the overall flow code coverage of the org falls below 90%, this will also show as a notification in the Health Inspector email alert
    - ***Extension Class***
        - If desired, an extension class can be referenced here which allows the data that is collected as part of the execution can be used in a post process
    - ***Administrator Email***
        - Enter the email address of your primary Salesforce administrator. This information will appear in the footer of the email message that is sent.
    - ***Administrator Name***
        - Enter the name of your primary Salesforce administrator. This information will appear in the footer of the email message that is sent.
    - ***Administrator Title***
        - Enter the title address of your primary Salesforce administrator. This information will appear in the footer of the email message that is sent.
    - ***Save File on Success***
        - If checked, when the app runs, it will create a date & time stamped Salesforce File which includes the contents of the success email that is generated on the specific execution
    - ***Save File on Failure***
        - If checked, when the app runs, it will create a date & time stamped Salesforce File which includes the contents of the failure email that is generated on the specific execution
    - ***Protected Component***
        - Leave the default value of false/unchecked
- **Failure Email Section**
    - ***Fail Email Subject***
        - Enter the Subject line of the email that is sent with the Health Inspector failure results
    - ***Send Email on Failure***

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved

        - Indicates whether an email should be sent if a failure is found
    - ***Fail Email Template***
        - Contains the HTML body of the failure email message that will be sent via the Health Inspector app
        - The 'Default' CMDT record contains a preconfigured template however this can be adjusted to meet your specific needs
- **Success Email Section**
    - ***Success Email Subject***
        - Enter the Subject line of the email that is sent with the Health Inspector success results
    - ***Send Email on Success***
        - Enter the Subject line of the email that is sent with the Health Inspector failure results
    - ***Success Email Template***
        - Contains the HTML body of the success email message that will be sent via the Health Inspector app
        - The 'Default' CMDT record contains a preconfigured template however this can be adjusted to meet your specific needs
- **Code Coverage Section (optional)**
    - ***Calculate Flow Code Coverage***
        - Indicates whether a calculation of flow code coverage should be done as part of the Health Inspector execution
        - **Important Note:** Requires a Connected App, Auth Provider & Named Credentials to be created to work as intended
            - **Creating a Connected App: *https://sforce.co/2D71HWd***
            - **Setup an Auth Provider: *https://sforce.co/333ZZQ3***
            - **Create Named Credentials: https://sforce.co/37mxpwL**
    - ***Calculate Code Coverage***
        - Indicates whether a calculation of code coverage should be done as part of the Health Inspector execution
        - **Important Note:** Requires a Connected App, Auth Provider & Named Credentials to be created to work as intended
            - **Creating a Connected App: *https://sforce.co/2D71HWd***
            - **Setup an Auth Provider: *https://sforce.co/333ZZQ3***
            - **Create Named Credentials: https://sforce.co/37mxpwL**
    - ***Named Credential***
        - Enter the name of the Named Credential that was created to support the Calculate Code Coverage functionality
        - Only required if using this functionality

Once all the information has been entered, click the Save button.

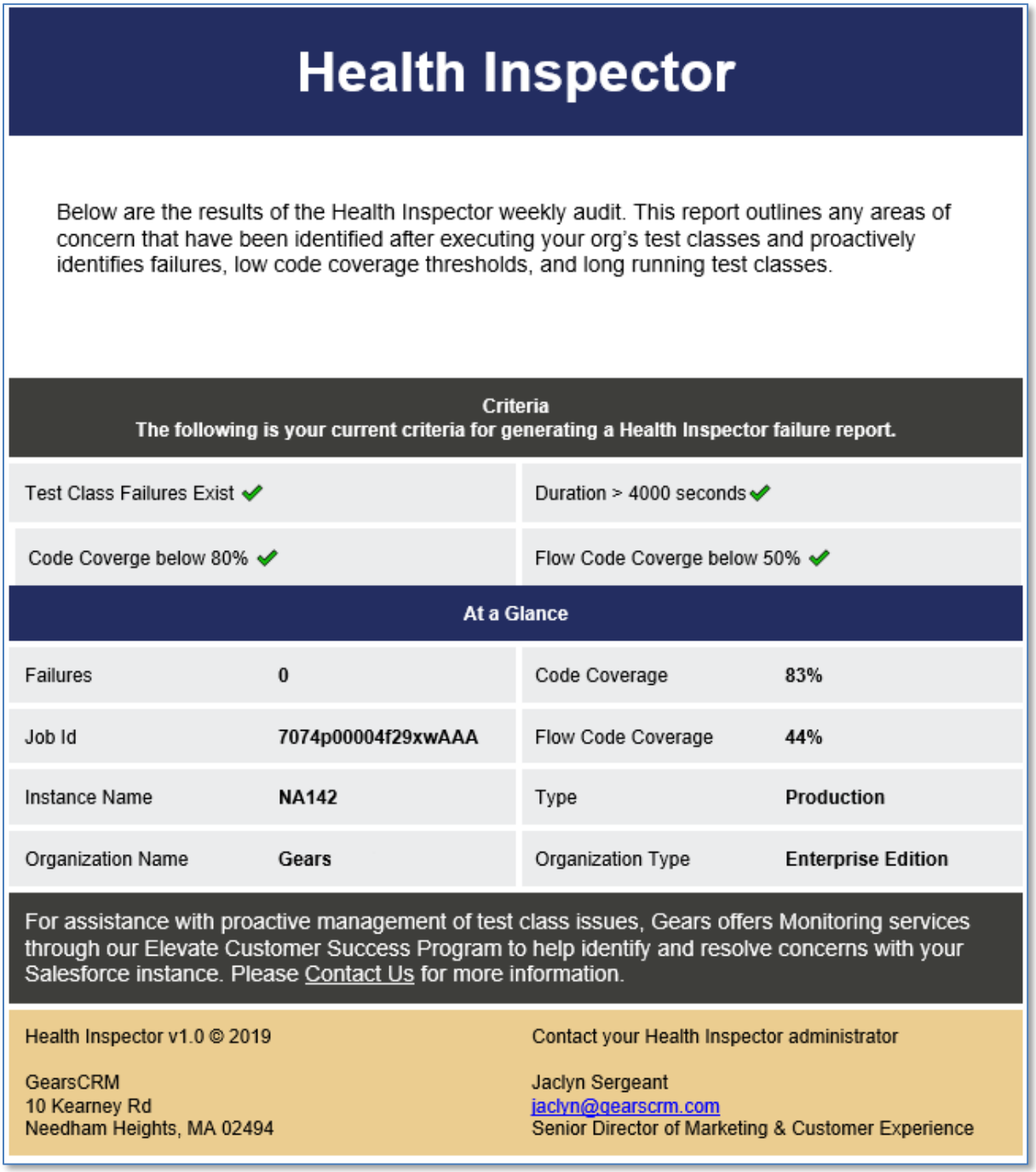
GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved



If you are creating more than one configuration for the Health Inspector app, simply repeat these steps until a custom metadata type record has been created for each individual scenario.

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved

Sample email result when using the 'DefaultWithFlow' option provided within the app:



Health Inspector

Below are the results of the Health Inspector weekly audit. This report outlines any areas of concern that have been identified after executing your org's test classes and proactively identifies failures, low code coverage thresholds, and long running test classes.

Criteria
The following is your current criteria for generating a Health Inspector failure report.

Test Class Failures Exist ✔	Duration > 4000 seconds ✔
Code Coverge below 80% ✔	Flow Code Coverge below 50% ✔

At a Glance

Failures	**0**	Code Coverage	**83%**
Job Id	**7074p00004f29xwAAA**	Flow Code Coverage	**44%**
Instance Name	**NA142**	Type	**Production**
Organization Name	**Gears**	Organization Type	**Enterprise Edition**

For assistance with proactive management of test class issues, Gears offers Monitoring services through our Elevate Customer Success Program to help identify and resolve concerns with your Salesforce instance. Please Contact Us for more information.

Health Inspector v1.0 © 2019

GearsCRM
10 Kearney Rd
Needham Heights, MA 02494

Contact your Health Inspector administrator

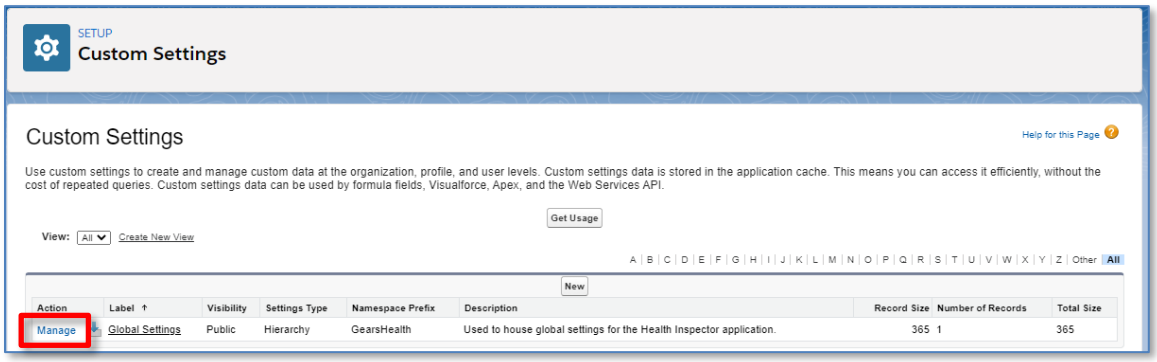
Jaclyn Sergeant
jaclyn@gearscrm.com
Senior Director of Marketing & Customer Experience

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved

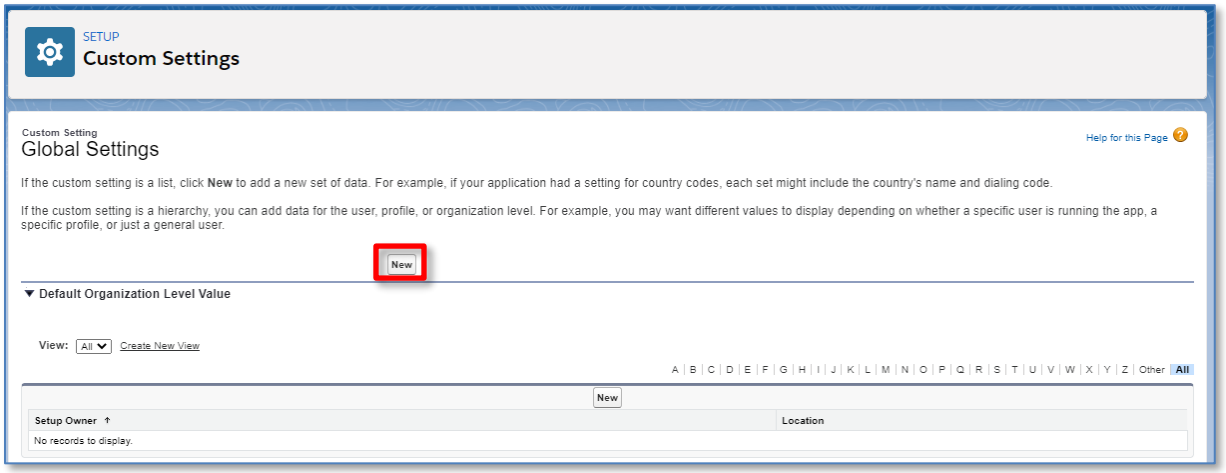
Global Settings (Global_Settings__c)

The 'Global Settings' Custom Setting is used to house the last successful run date/time and status of the Health Inspector application.

The application will automatically create a Custom Setting record on its initial run however if you'd like to set initial values, navigate to the Salesforce Setup menu and type in 'Custom Settings' in the search box and select that from the menu. In the resulting screen, click on the Manage link to the left of the 'Global Settings' entry:

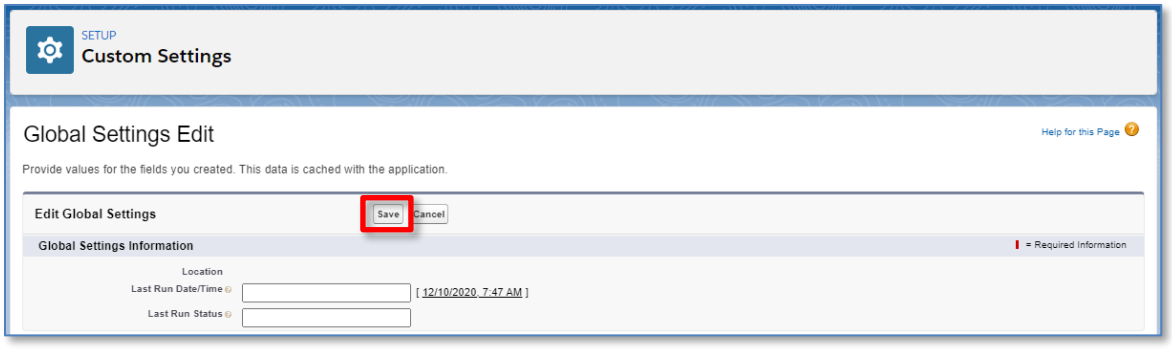


Next, click on the 'New' button at the top of the screen. Do not click the 'New' button found right above the list view as these two buttons result in varying setups.



After clicking 'New', you will be brought to a screen that shows a 'Last Run Date/Time' field and a 'Last Run Status' field. You can leave both fields null and just click the 'Save' button.

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved



As you finalize the Health Inspector setup and the app runs, the values from your execution will write back to this Custom Setting. In doing so, those values will then be shown on the 'Health Inspector Results' tab found within the application.

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved

Scheduling Health Inspector

> ***Important Note:***
>
> If you are using either the 'Default' or 'DefaultWithFlow' Test Class Audit Parameter custom metadata types that are delivered with the application, you can skip down to the 'Create Schedule' step.
>
> Otherwise, if you've created a custom Test Class Audit Parameter entry, follow the 'Create Custom Class to Schedule' step and then 'Create Schedule' step.

Create Custom Class to Schedule

In order to schedule the Health Inspector application to run on a custom CMDT entry:

- Go to the Developer Console → File → New → Apex Class
- Enter a name for the class
    - For example, if you're CMDT entry is called 'MyTest', then perhaps give it a name of 'HealthInspector_MyTest'
- Click OK
- Overwrite the default class text with the following:

```
global class HealthInspector_MyTest implements Schedulable
{
   global void execute(SchedulableContext SC)
   {
            GearsHealth.initiateAudit.initiateApexTestExecution('Default');
   }
}
```

*The "HealthInspector_MyTest' value should be replaced with the name you gave the class in the second step
**The "Default" value in the above snippet should be replaced with the name of the CMDT that was configured as part of the setup process.

Create Schedule

To schedule the 'Default' or 'DefaultWithFlow' entry:

- Go to the Salesforce Setup menu
- Search for 'Apex Classes' in the left-hand navigation and select it
- Click on the 'Schedule Apex' button
- From the resulting screen, enter the following information:
    - Job Name = <Enter intuitive name for the schedule>

GearsDesign
Copyright© 2021 GearsDesign, Inc. All Rights Reserved

- Class Name
  - If you are using the 'Default' CMDT, then select ***HealthInspectorScheduler_Default***
  - Else, if you are using the 'DefaultWithFlow' CMDT, then select ***HealthInspectorScheduler_DefaultWithFlow***
  - Otherwise, you'll want to select the class that was defined/deployed in the 'Create Custom Class to Schedule' section.
- Select the appropriate frequency for execution
- Click Save

GearsDesign

Copyright© 2021 GearsDesign, Inc. All Rights Reserved

Additional Configuration Options

There are a few additional configuration options available in the Health Inspector application – executing the batch manually, implementing an extension class and re-running the last test that was generated.

Executing Batch Manually

In some instances, you may want to manually execute the Health Inspector app to verify your configuration before putting it on a scheduled frequency. In order to do this, simply go to the Developer Console → Debug → Open Execute Anonymous Window and paste the following:

```
GearsHealth.initiateAudit.initiateData i =
GearsHealth.initiateAudit.initiateApexTestExecutionReturn(CMDT Name);

system.debug('Error Message = ' + i.errorMessage);
system.debug('Job Ids = ' + i.jobIdList);
```

*The "CMDT Name" value in the above snippet should be replaced with the name of the CMDT that was configured as part of the setup steps.

Implement Extension Class

In some instances, you may want to take the data collected by the Health Inspector application and use that in a secondary process. The application allows you to do this by referencing an Extension Class which is configured in the Test Class Audit Parameter CMDT. The Extension Class should reference the processResults method to return this information. For example:

```
global with sharing class Extension_Example implements
GearsHealth.HealthInspectorExtension
{
   global object processResults(id jobId, string
auditParameterName)
   {
      system.debug('SAMPLE DEBUG');
      return true;
   }
}
```

*The "SAMPLE DEBUG" value in the above snippet can be replaced with any debug information that should be written out at the time of its execution

GearsDesign

Copyright© 2021 GearsDesign, Inc. All Rights Reserved

Re-Run Last Test Generated

In some instances, you may want to re-rerun the last test that was generated. In order to do this, you can simply run the following snippet via the Developer Console:

```
AsyncApexJob job = [Select id From AsyncApexJob
where jobType = 'TestRequest'
order by createddate desc limit 1];

// this assumes 1 job Id
List <id> jobIdList = new List <id> {job.id};

gearsHealth.sendEmailv2.generateEmail(jobidList, 'Default');
```

*The “Default” value in the above snippet should be replaced with the name of the CMDT that was configured as part of the setup process.

GearsDesign

Copyright© 2021 GearsDesign, Inc. All Rights Reserved

Best Practices

In some cases, you may find that the Health Inspector executions result in a 'UNABLE_TO_LOCK_ROW' error which is caused by tests attempting to run in parallel and ultimately caused by record contention between more than one test. To alleviate this issue, follow the steps found at the following link under the 'Best Practices for Parallel Test Execution' section:

- https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_testing_best_practices.htm

GearsDesign

Copyright© 2021 GearsDesign, Inc. All Rights Reserved