[Document History]

- 2019.12.13 Create this Document. (v1.0)
-

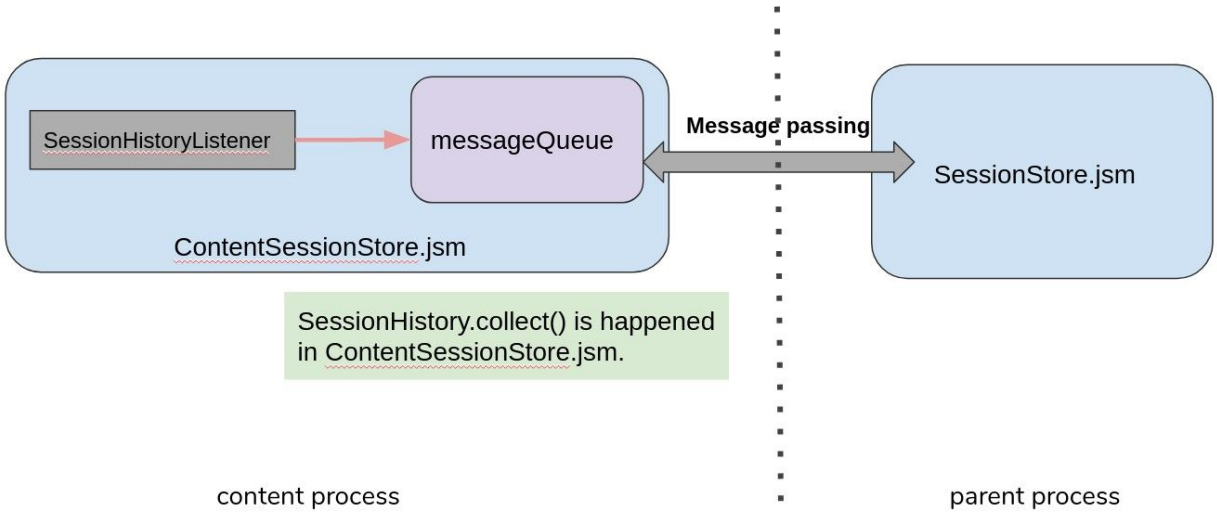
[Content]

[Main idea]

- **Reduce the memory usage of content process.**
  - When enabling fission, we will have more content processes than before. The memory usage of content process becomes more and more important.
  - ContentSessionStore.jsm is currently loaded into every tab frameloader. We can achieve the goal by rewriting the code into C++.
    - Rewritings for the modules loaded in ContentSessionStore.jsm.
      - Bug 1497144 - DocShellCapabilities.jsm and ScrollPosition.jsm
      - Bug 1497146 - FormData.jsm
      - Bug 1497147 - Utils.jsm (rewrite mapFrameTree)
      - Bug 1507286 - SessionStorage.jsm
    - Rewritings for the listener in the ContentSessionStore.jsm:
      - Bug 1474130 - ScrollPosition/Privacy/DocCapability listeners
      - Bug 1544371 - FormDataListener
      - Bug 1549975 - SessionStoreListener
- **Make the sessionRestore functionality work when session history living in the parent process.**
  - Fission team is designing the new generation of session history. First idea is move session history from the child process to the parent process.
  - Now we can enable sessionHistory in the parent process by pref "**fission.sessionHistoryInParent**".
  - Related bug: Bug 1438272

[Current architecture for collect sessionHistory]

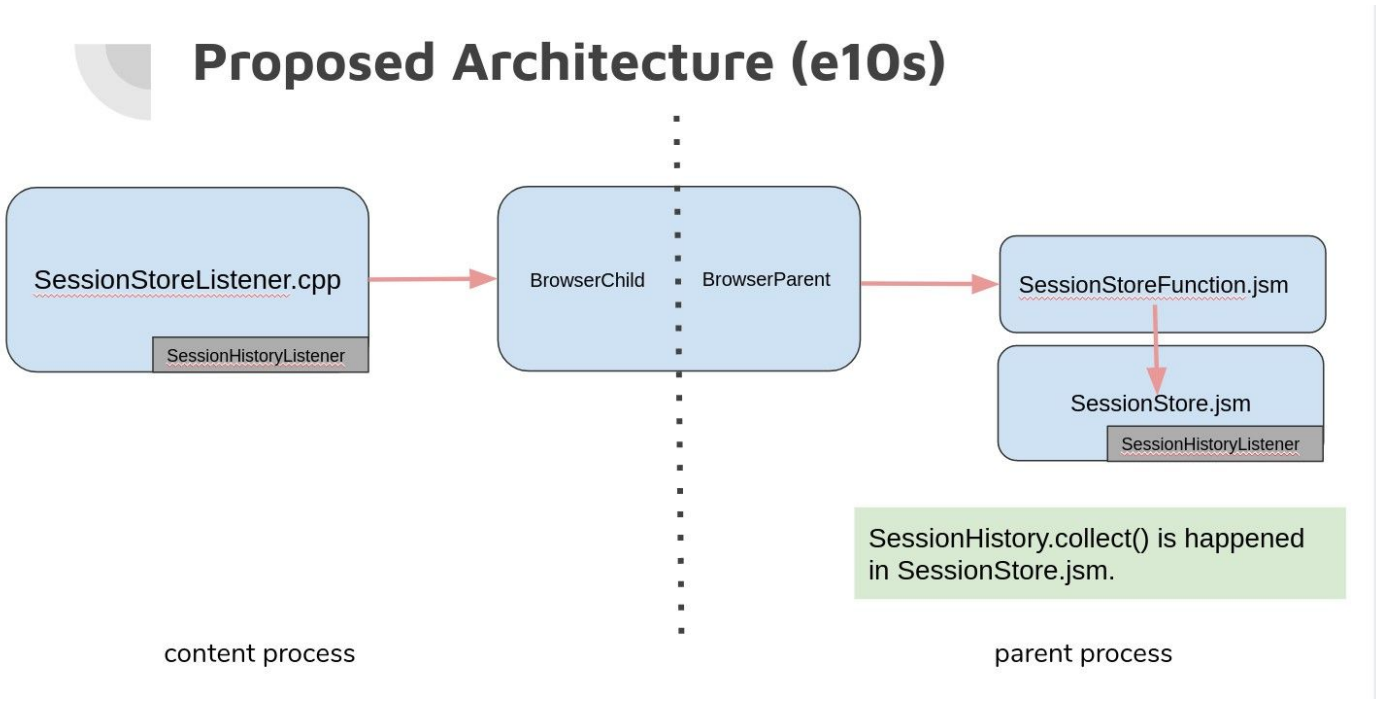
- There is a sessionHistoryListener which is running in the content process.
  - The listener is implemented in ContentSessionStore.jsm. [source code]
  - We use the listener to listen for changes of sessionHistory. Whenever the user navigates we will collect everything needed for session history. ContentSessionStore will use "**SessionStore:update**" message to update the data we collect in that listener.
  - How we collect everything needed for sessionHistory?
    - By SessionHistoryListener.collect() and use SessionHistory module to collect the sessionHistory.



-

[Proposed architecture for collect sessionHistory]

1. Move the original implementation of “sessionHistoryListener” from ContentSessionStore.jsm into a new JSM module(SessionHistoryListener.jsm).
2. When the pref **“fission.sessionHistoryInParent” is false**:
    - Load “SessionHistoryListener.jsm” and new a SessionHistoryListener as before. The behavior keeps the same.
3. When the pref **“fission.sessionHistoryInParent” is true**:
    - Don’t load “SessionHistoryListener.jsm” to reduce memory usage.
    - Since ‘sessionHistory’ object should be accessed in the parent process, we divided the listener into 2 places.
        i. sessionHistoryListener in the parent process is in charge of everything related to sessionHistory object.
            - We collect everything needed for sessionHistory here. In order to use SessionHistory module to collect the sessionHistory, we will refactor the related function as well.
        ii. sessionHistoryListener in the child process(SessionStoreListener.cpp) is in charge of the other events we need to listen in the child process.
    - The sessionStore update flow is also changed. We use the same mechanism as Bug 1474130, Bug 1544371 and Bug 1549975.
        i. Use the update function in the “SessionStoreFunction.jsm” to update data into SessionStore.jsm directly.

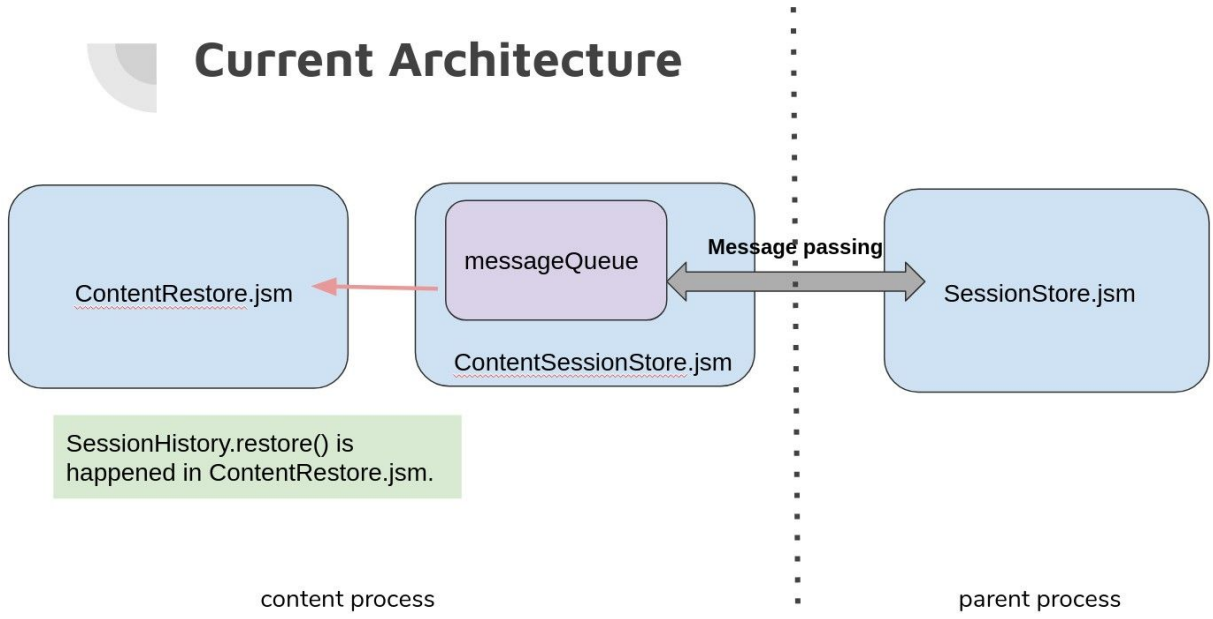


If sessionHistory update is needed,

- Use **nsFrameLoader::RequestSHistoryUpdate** to notify SessionStoreListener that there is shistory changes.
- In the next sessionStore update, SessionStore.jsm will do **SessionHistory.collectFromParent()**.
- Before collecting session history changes, we only need to request once.

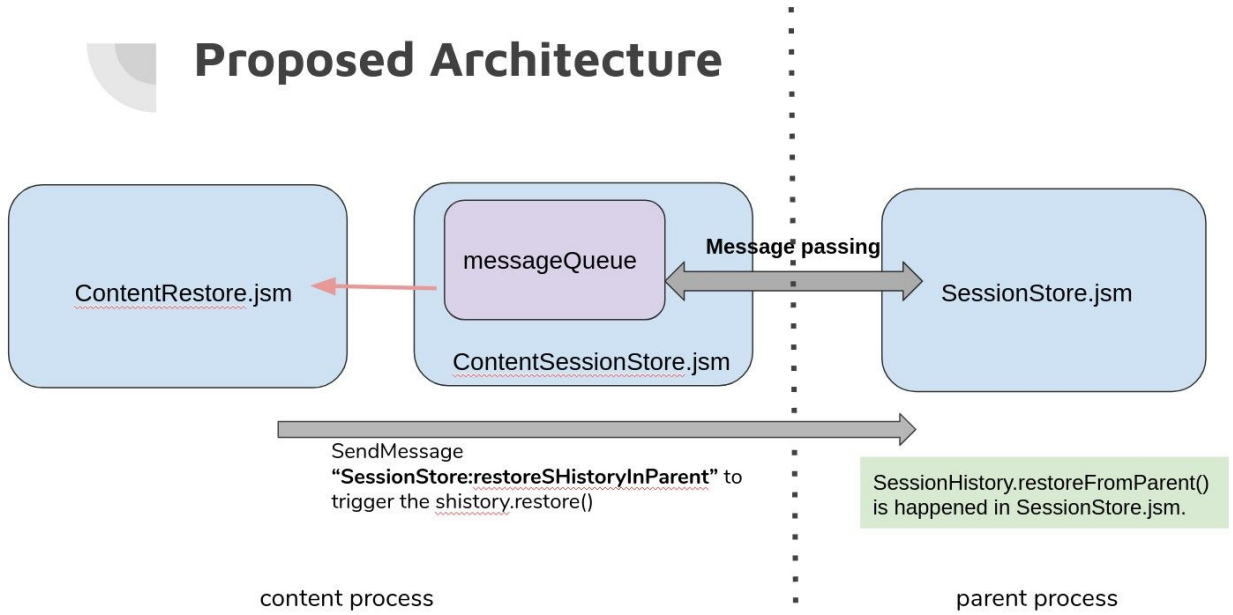
[Current architecture for restore sessionHistory]

- When receiving restore message from sessionStore.jsm, contentSessionStore will use ContentRestore to do the restore.



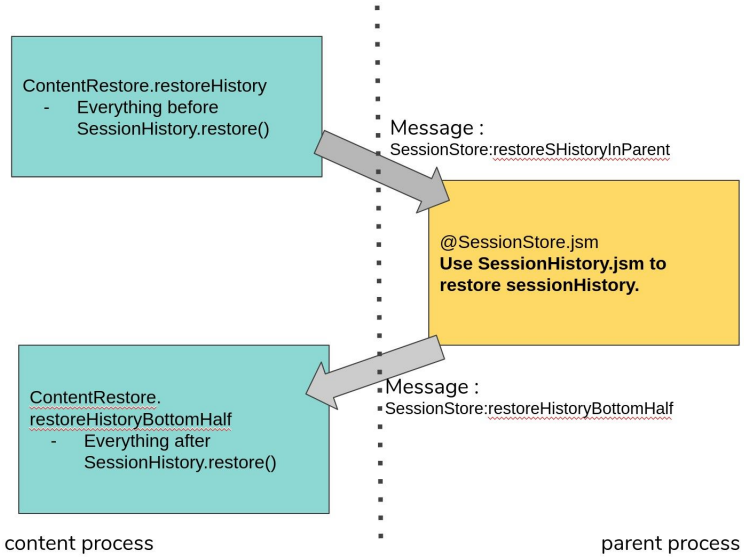
[Proposed architecture for restore sessionHistory]

- SendMessage to parent process to do the session history restore.



(Current)

(Proposed Architecture when pref on)



[New added messages]

From child to parent	
SessionStore:AddSHistoryListener	In contentSessionStore constructor, we need to create the listener for session history. If pref on, we need to ask sessionStore to create corresponding listener in the parent process.
SessionStore:removeRestoreListener	When doing restore, we need to add another listener for different purposes in the parent process. Use this message to notify sessionStore when the listener is not needed anymore.
SessionStore:restoreSHistoryInParent	When needing to restore session history with pref on, use this message to ask sessionStore to do the restore.
SessionStore:reloadCurrentEntry	When needing to do the current history entry with pref on, use this message to ask sessionStore to do it.

From parent to child	
SessionStore:restoreHistoryBottomHalf	After restoring session history in the parent, use this message to notify ContentSessionStore.
SessionStore:OnHistoryReload	The message is called in the callback of the listener we added when doing the restore.
SessionStore:OnHistoryNewEntry	The message is called in the callback of the listener we added when doing the restore.

[Follow-up Bug after Bug 1507287]

There are some known mochitests failures when enabling sessionHistory in the parent process.

- Bug 1599105 for **browser_500328.js**.
- bug 1602486 for **browser_scrollPositions.js**
- Bug 1602501 for **browser_swapDocShells.js**.