/\*\*

\* =============================================================================

\* @file ai\_dialogue.gs

\* @module GAssistant.AI.Dialogue

\* @description

\* هذه الوحدة مسؤولة عن توحيد تنسيق الرسائل التي تظهر للمستخدم في الواجهة الأمامية

\* (مثل الرسائل الجانبية أو رسائل الخطأ)، بالإضافة إلى إدارة سجل المحادثة.

\* تعتمد على GAssistant.AI.Memory و GAssistant.Utils.

\* =============================================================================

\*/

const GAssistant = GAssistant || {};

GAssistant.AI = GAssistant.AI || {};

/\*\*

\* GAssistant.AI.Dialogue: وحدة إدارة الحوار.

\*

\* @param {object} memory - كائن إدارة الذاكرة (GAssistant.AI.Memory).

\* @param {object} utils - كائن الأدوات المساعدة العامة (GAssistant.Utils).

\*/

GAssistant.AI.Dialogue = ((memory, utils) => {

// ---------------------------------------------------------------------------

// دوال مساعدة داخلية

// ---------------------------------------------------------------------------

/\*\*

\* @function \_createMessageObject

\* @description

\* ينشئ كائن رسالة موحدًا للواجهة الأمامية.

\* @param {string} type - نوع الرسالة ('success', 'error', 'info', 'warning').

\* @param {string} text - نص الرسالة.

\* @returns {object} - كائن الرسالة.

\*/

function \_createMessageObject(type, text) {

return {

type: 'message',

level: type, // 'success', 'error', 'info', 'warning'

text: text

};

}

// ---------------------------------------------------------------------------

// الدوال العامة (Public)

// ---------------------------------------------------------------------------

/\*\*

\* @function showMessage

\* @description

\* تُنشئ رسالة معلوماتية للواجهة الأمامية.

\* @param {string} text - نص الرسالة.

\* @returns {object} - كائن الرسالة.

\*/

function showMessage(text) {

utils.log(`[AI.Dialogue] Info: ${text}`);

return \_createMessageObject('info', text);

}

/\*\*

\* @function showSuccess

\* @description

\* تُنشئ رسالة نجاح للواجهة الأمامية.

\* @param {string} text - نص الرسالة.

\* @returns {object} - كائن الرسالة.

\*/

function showSuccess(text) {

utils.log(`[AI.Dialogue] Success: ${text}`);

return \_createMessageObject('success', text);

}

/\*\*

\* @function showError

\* @description

\* تُنشئ رسالة خطأ للواجهة الأمامية.

\* @param {string} text - نص الرسالة.

\* @returns {object} - كائن الرسالة.

\*/

function showError(text) {

utils.error(`[AI.Dialogue] Error: ${text}`);

return \_createMessageObject('error', text);

}

/\*\*

\* @function showWarning

\* @description

\* تُنشئ رسالة تحذير للواجهة الأمامية.

\* @param {string} text - نص الرسالة.

\* @returns {object} - كائن الرسالة.

\*/

function showWarning(text) {

utils.warn(`[AI.Dialogue] Warning: ${text}`);

return \_createMessageObject('warning', text);

}

/\*\*

\* @function showSummaryTable

\* @description

\* تُنشئ كائنًا لعرض جدول ملخص في الواجهة الأمامية.

\* @param {Array<Array<any>>} rows - مصفوفة من الصفوف، كل صف هو مصفوفة من القيم.

\* @param {string} title - عنوان الجدول.

\* @param {Array<string>} headers - رؤوس الأعمدة.

\* @returns {object} - كائن الجدول.

\*/

function showSummaryTable(rows, title, headers) {

if (!Array.isArray(rows) || !Array.isArray(headers)) {

utils.error('[AI.Dialogue] Invalid input for showSummaryTable.');

return showError('فشل عرض الجدول: بيانات غير صالحة.');

}

utils.log(`[AI.Dialogue] Displaying summary table: ${title}`);

return {

type: 'summary\_table',

content: title,

headers: headers,

rows: rows

};

}

/\*\*

\* @function showSuggestions

\* @description

\* تُنشئ كائنًا لعرض قائمة اقتراحات في الواجهة الأمامية.

\* @param {Array<object>} suggestions - مصفوفة من كائنات الاقتراحات ({description: string}).

\* @param {string} title - عنوان قسم الاقتراحات.

\* @returns {object} - كائن الاقتراحات.

\*/

function showSuggestions(suggestions, title) {

if (!Array.isArray(suggestions)) {

utils.error('[AI.Dialogue] Invalid input for showSuggestions.');

return showError('فشل عرض الاقتراحات: بيانات غير صالحة.');

}

utils.log(`[AI.Dialogue] Displaying suggestions: ${title}`);

return {

type: 'suggestions',

content: title,

suggestions: suggestions

};

}

/\*\*

\* @function summarizeSession

\* @description

\* تلخص سجل التفاعل الأخير من الذاكرة وتقدمه كنص.

\* @returns {object} - كائن نصي يحتوي على ملخص الجلسة.

\*/

function summarizeSession() {

const recentInteractions = memory.getRecentInteractions();

let summaryText = "ملخص الجلسة:\n";

if (recentInteractions.length === 0) {

summaryText += "لا توجد تفاعلات حديثة في الذاكرة.";

} else {

recentInteractions.forEach(item => {

summaryText += `- [${item.timestamp}] ${item.type}: ${JSON.stringify(item.data).substring(0, 100)}...\n`;

});

}

utils.log('[AI.Dialogue] Session summarized.');

return {

type: 'text',

content: summaryText

};

}

/\*\*

\* @function repeatLastPrompt

\* @description

\* يجلب آخر طلب مستخدم من الذاكرة.

\* (هذه الدالة قد تحتاج إلى إعادة هيكلة لاستدعاء AI.Core.ask مباشرة)

\* @returns {object} - كائن نصي يحتوي على آخر طلب.

\*/

function repeatLastPrompt() {

const lastPrompt = memory.getLastUserPrompt();

if (lastPrompt) {

utils.log(`[AI.Dialogue] Repeating last prompt: ${lastPrompt}`);

return {

type: 'text',

content: `آخر طلب: "${lastPrompt}"`

};

} else {

utils.log('[AI.Dialogue] No last prompt to repeat.');

return showMessage('لا يوجد طلب سابق لإعادته.', 'info');

}

}

/\*\*

\* @function presentDecisionWithJustification

\* @description

\* تُنشئ كائنًا لطلب تأكيد من المستخدم قبل تنفيذ إجراء حرج.

\* @param {string} proposedAction - وصف الإجراء المقترح.

\* @param {string} justification - المنطق أو التبرير وراء الاقتراح.

\* @param {object} payload - الحمولة التي سيتم تمريرها إذا وافق المستخدم.

\* @returns {object} - كائن طلب التأكيد.

\*/

function presentDecisionWithJustification(proposedAction, justification, payload) {

utils.log(`[AI.Dialogue] Presenting decision for confirmation: ${proposedAction}`);

return {

type: 'confirmation\_required',

message: `هل أنت متأكد من أنك تريد: ${proposedAction}\nالمنطق: ${justification}`,

payload: payload

};

}

// ---------------------------------------------------------------------------

// واجهة الوحدة (Public API)

// ---------------------------------------------------------------------------

return {

showMessage: showMessage,

showSuccess: showSuccess,

showError: showError,

showWarning: showWarning,

showSummaryTable: showSummaryTable,

showSuggestions: showSuggestions,

summarizeSession: summarizeSession,

repeatLastPrompt: repeatLastPrompt,

presentDecisionWithJustification: presentDecisionWithJustification

};

})(

GAssistant.AI.Memory, // حقن GAssistant.AI.Memory

GAssistant.Utils // حقن GAssistant.Utils

);