内置节点目录

本文汇总当前版本内置节点的功能、配置与输出，便于在 FlowBuilder 中复用或扩展。所有节点都暴露统一的触发机制：

- **输入触发器** `trigger`（必填）：用于接收上游节点发出的执行信号。
- **输出触发器** `trigger`（必填）：节点完成后发出的继发信号，可推动下游节点继续运行。

LoadTable

- **类型**：`LoadTable`
- **分类**：table
- **输入变量**：
  - `trigger`：外部事件触发器
- **配置**：
  - `sheetId?: string` 指定工作表 ID（默认使用 `FlowContext.sheetId`）
- **输出变量**：
  - `trigger`：加载完成后发出的继发信号
  - `snapshot`：最新表格快照
  - `variables`：写入 `sheetId`、`lastSnapshot`
- **失败场景**：缺少 sheetId 或适配器未注入

ParseTags

- **类型**：`ParseTags`
- **分类**：ai
- **输入变量**：
  - `conversation?`：`FlowContext.conversation`（可选）
  - `trigger`：外部事件触发器
- **配置**：无
- **输出变量**：
  - `trigger`：解析完成后发出的继发信号
  - `instructions`：`TagInstruction[]`
  - `variables`：写入解析结果
  - `variables.instructions`：写入 FlowContext 供下游节点引用
- **说明**：默认解析器识别 `[[table:action key=value]]` 语法，可通过 `runtime.getParser().register()` 扩展

ApplyChanges

- **类型**：`ApplyChanges`
- **分类**：table
- **输入变量**：

- `instructions`：`TagInstruction[]`
- `snapshot?`：最新快照（可选）
- `trigger`：外部事件触发器

- **配置**：
  - `sheetId?: string` 覆盖目标表
  - `dryRun?: boolean` 仅返回命令，不提交
  - `instructionsKey?: string` 指定变量路径（默认 `variables.instructions`）

- **输出变量**：
  - `trigger`：执行完成后发出的继发信号
  - `commands` / `preview`：生成的命令数组
  - `snapshot`：变更后的快照（非 dry-run）
  - `variables`：写入 `lastSnapshot`、`lastCommands`
  - `variables.lastSnapshot` / `variables.lastCommands`

- **指令映射**：
  - `setCell` / `set-cell` → `type: 'set'`
  - `deleteCell` / `delete-cell` → `type: 'delete'`
  - `insertRow` / `insert-row` → `type: 'insertRow'`
  - `removeRow` / `remove-row` → `type: 'removeRow'`
  - `insertColumn` / `insert-column` → `type: 'insertColumn'`
  - `removeColumn` / `remove-column` → `type: 'removeColumn'`

- **注意事项**：
  - 若指令未指定 `sheet` 字段，则使用配置或上下文中的 sheetId
  - `dryRun` 会在执行后重置 staged changes，不会提交

单元格/行/列命令详解

命令 `type`	适用指令	载荷字段	说明
`set`	`setCell` / `set-cell`	`row: number`, `column: number`, `value?: unknown`, `meta?: Record<string, unknown>`	写入或覆盖指定单元格，可携带元数据
`delete`	`deleteCell` / `delete-cell`	`row: number`, `column: number`	清空单元格内容，`value` 会被置为 `null`
`insertRow`	`insertRow` / `insert-row`	`index: number`, `values?: unknown[]`, `meta?: Record<string, unknown>`	在 `index` 位置前插入一行，可附带默认行值与元信息

sudo apt install net-tools

查看所有接口

ifconfig

只显示活跃接口

ifconfig | grep "inet "

查看特定接口

ifconfig eth0
`removeRow`	`removeRow` / `remove-row`	`index: number`	删除指定行
`insertColumn`	`insertColumn` / `insert-column`	`index: number`, `values?: unknown[]`, `meta?: Record<string, unknown>`	在 `index` 位置前插入一列，可附带列默认值
`removeColumn`	`removeColumn` / `remove-column`	`index: number`	删除指定列

> 提示：如需一次处理多个区域，可在解析阶段拆分指令数组，或直接在自定义节点中批量构造多个命令后交给 `TableAdapter.applyChanges`。

FormatPrompt

- **类型**：`FormatPrompt`
- **分类**：formatters
- **输入变量**：
  - `snapshot?`：待格式化的快照（可选）
  - `instructions?`：指令数组，供格式化器参考（可选）
  - `trigger`：外部事件触发器
- **配置**：
  - `formatter?: string` 默认 `prompt`
  - `selectFrom?: string` 指定变量路径，覆盖 `variables.lastSnapshot`
- **输出变量**：
  - `trigger`：格式化完成后发出的继发信号
  - `formatted`：格式化结果
  - `variables`：写入最新格式化内容
  - `variables.lastFormatted`
- **内置格式化器**：
  - `prompt`：以多行文本形式输出 sheet、revision、metadata 与逐行内容
  - `json`：输出 `{ sheetId, revision, metadata, rows }`

注册内置节点

```
import { NodeRuntime, registerBuiltinNodes } from '@table-git/memory-engine';
```

```
const runtime = new NodeRuntime({ adapter });
registerBuiltinNodes(runtime);
```

该函数会：

1. 注册上述四个节点
2. 注册默认格式化器 `prompt`、`json`
3. 注册基础标签解析器 `tag-bracket-parser`

自定义节点最佳实践

1. **Schema 描述**：在 `getSchema` 中补充 label、summary、category，便于未来 UI 渲染。
2. **配置校验**：通过 `validate` 返回直观错误，避免运行时才失败。
3. **变量管理**：将共享数据写入 `flowContext.variables`，或通过 `state.set(node.id, outputs)` 返回局部结果。
4. **事件与日志**：利用 `services.events.emit` 与 `services.log` 上报关键动作。
5. **可测试性**：独立编写单元测试，模拟 `NodeExecutionContext` 以覆盖边界状况。

TODO 与计划

- 查询类节点：根据条件读取表格数据并写入变量
- 冲突解决节点：对 diff/冲突进行可配置合并
- 事件节点：在事件触发时调用 webhook 或函数
- 条件路由节点：基于变量控制执行路径

欢迎根据业务需求扩展节点，并通过 `NodeRuntime.register` 注入。