docs_chain

.لا يمكن لأي مستند مُتعقَّب أن يخرج عن التزامن — مُجزَّأ المحتوى، ثنائي الاتجاه، ذري

الملخص

Docs Chain هو محرك شامل لتنفيذ التبعيات ثنائية الاتجاه بين المستندات وقواعد البيانات، أو Markdown عند تغيير أي عنصر في سلسلة مسجلة — سواء كان مصدر Go. يعتمد على قطع مبكر، حواف مزامنة، Kahn عبر رسم بياني موجه (ترتيب طوبولوجي Salsa المحتوى بأسلوب ثنائية الاتجاه، عمليات إعادة تسمية ذرية + التزام معاملات SQLite)، يقوم بإعادة توليد التصديرات HTML/PDF/DOCX أو قاعدة بيانات SQLite — يقوم المحرك باكتشاف تصدير من نوع التغيير عبر تجزئة المحتوى وينشره عبر جميع العناصر المتصلة بشكل ذري.

وصف مختصر

محرك Go يحافظ على تزامن المستندات وقواعد البيانات. باستخدام إعادة الحساب التزايدي المُجزَّأ المحتوى بأسلوب Salsa عبر رسم بياني موجه (ترتيب طوبولوجي Kahn، حواف مزامنة قطع مبكر، عمليات إعادة تسمية ذرية + التزام معاملات SQLite)، ثنائية الاتجاه، يقوم بإعادة توليد التصديرات عند تغيير أي عنصر مرتبط.

وصف مفصل

Docs Chain هو ما تبنيه بعد كتابة نفس السكربت الهش "أعد توليد PDF عند تغيير مرة واحدة أكثر من اللازم. يحل هذا المحرك محل فئة كاملة من أدوات المزامنة Markdown" اليدوية بمحرك حقيقي. فهو يعامل مستندات المشروع وقواعد البيانات كأعضاء في سلسلة، وعند تغيير أي عضو، يقوم بنشر هذا التغيير عبر جميع الأعضاء المتصلين في كل اتجاه مُعلن عنه — لإعادة التوليد والتصدير بشكل ذري لضمان عدم خروج أي عنصر متعقب عن التزامن، أبداً. يعتمد التصميم على الصرامة المستمدة من أنظمة البناء التزايدي بدلاً من السكربتات: يتم اكتشاف التغييرات عبر **تجزئة المحتوى**، لا يُطلق أي شيء، بينما يؤدي تعديل بحرف واحد إلى إعادة `touch` لذا فإن أمر **وليس وقت التعديل**، البناء بالضبط لما يجب إعادة بنائه — دون إنذارات كاذبة أو تغييرات مفقودة. وبصياغة رسمية في سطر واحد، فهو إعادة حساب تزايدي مُجزَّأ المحتوى بأسلوب Salsa عبر رسم بياني موجه، مع ترتيب طوبولوجي Kahn، وحواف مزامنة ثنائية الاتجاه ذات قطع مبكر يتجاهل الفروع الفرعية دون تغيير، وحواف مزامنة ثنائية الاتجاه ذات سلطة مُعلن عنها، بالإضافة إلى إعادة تسمية ذرية والتزام معاملات SQLite لضمان عدم ترك أي تصدير نصف مكتوب في حال حدوث عطل أثناء النشر. يُوزَّع المحرك كوحدة فرعية لـ `vasic-digital` ويُستخدم كجزء أساسي من الوحدة الفرعية HelixConstitution، بحيث يحصل أي مشروع يتبنى الدستور على Docs Chain عبر سلاسله الخاصة تلقائياً ويسجل YAML لكل سياق. يُظهر التنفيذ حالته بوضوح (وفقاً للمادة 11.4.6 من الدستور): تم تنفيذ واختبار المراحل 1–4 (الرسم البياني الموجه الأساسي + التجزئة، محولات/تحولات العقد، منسق النشر مع الذرية)، CLI ؛ تضيف `sync/verify/doctor/graph/watch` متعدد السياقات مدفوع بالتكوين مع أوامر Go، نقية) `md-to-sqlite/sqlite-to-md` أدوات مدمجة ثنائية الاتجاه مثل المرحلة 4b

؛ تم تنفيذ`colorize-html` انحراف على مستوى الصف، رحلة ذهاب وإياب مستقرة بالبايت) وأداة المرحلة 5 (اختبار شامل ثنائي النهاية في العالم الحقيقي) وهي خضراء. تبقى المراحل 6–7 (توزيع هو أول مستهلك حقيقي Herald. مُخططاً لها ومقيدة بالمشغل (ATMOSphere الدستور، توصيل في المصب، حيث يقوم بمزامنة مجموعة متعددة التنسيقات تتألف من 66 مستنداً تحقق نظافة التزامن.

لماذا قمنا ببنائه

تنفصل الوثائق والتصديرات وقواعد البيانات عن بعضها لحظة صيانتها يدوياً أو عبر سكربتات هشة. يجعل المزامنة آلية، دقيقة عبر تجزئة المحتوى، وذرية، بحيث يؤدي التغيير في أي مكان من Docs Chain السلسلة إلى تحديث كل شيء في المصب (والأعلى) بشكل صحيح وآمن.

المحتوى

لماذا يُعدّ ثورةً في المجال

يستعير هذا النظام الضمانات الدقيقة التي يأخذها مؤلفو المترجمات وأنظمة البناء كأمر مسلّم به — مثل الرسوم البيانية للتبعيات المعتمدة على تجزئة المحتوى، وإعادة الحساب بأقل جهد ممكن، والالتزامات والنوايا cron الذرية — ويوجّهها نحو مجال الوثائق وقواعد البيانات، الذي ظلّ يعتمد تاريخياً على مهام الحسنة. يضمن المزامنة الثنائية الحقيقية فرض العلاقة بين المصدر وصادره في كلا الاتجاهين، بحيث تتحوّل عبارة "الوثائق قديمة" و"الصادر لا يتطابق مع المصدر" من أخطاء متكررة إلى حالات لن يسمح المحرّك بوجودها أصلاً.

ما الجديد؟

- إعادة حساب تدريجية معتمدة على تجزئة المحتوى (وليس وقت التعديل) عبر رسم بياني موجه غير دوري مع إيقاف مبكر.
- حواف مزامنة ثنائية الاتجاه ذات سلطة معلنة (الصادرات ↔ الوثائق ↔ SQLite).
- التزام ذري بإعادة تسمية + معاملة SQLite لضمان انتشار آمن حتى في حال الأعطال.
- مع كشف الانحراف `md-to-sqlite/sqlite-to-md` من Go رحلة ذهاب وإياب خالصة بـ على مستوى الصفوف.

التحديات والحلول

- **إعادة بناء غير ضرورية:** تم حلّها بالكشف عن تجزئة المحتوى بدلاً من الطوابع الزمنية.
- **تحديثات جزئية أو فاسدة:** تم حلّها بإعادة التسمية الذرية ومعاملات SQLite.
- **ترتيب الأعضاء المتعدد الصحيح:** تم حلّه بترتيب كان الطوبولوجي مع إيقاف مبكر.
- **الإبلاغ الصادق عن القدرات:** تم حلّه بوضع علامة على كل مرحلة بأنها مُنفّذة أو مُخطط لها وفقاً للفقرة 11.4.6.

المكدس التقني (السبب وكيفية العمل)

- **Go** — المحرّك بأكمله (`internal/hash`، `graph`، `adapter`، `orchestrator`، `config`، `state`، `runner`، `cmd/docs_chain`).

- **الرسم البياني الموجه غير الدوري + ترتيب كان الطوبولوجي** — ترتيب التبعيات مع إيقاف مبكر.
- **SQLite (نسخة خالصة من Go بمكتبة modernc)** — الأعضاء في قاعدة البيانات والالتزامات المعاملاتية.
- **fsnotify** — خادم `watch` للبث المباشر.
- **تكوين YAML** — تسجيل السلاسل حسب السياق.
- **exec: التحويلات** — توليد قابل للتوصيل من Markdown إلى HTML/PDF/DOCX.

> ATMOSphere) توزيع الدستور، توصيل) الصراحة في خارطة الطريق: المرحلتان 6–7 مُخطط لهما / محكومة بالتشغيل — ولم تُصدر بعد.