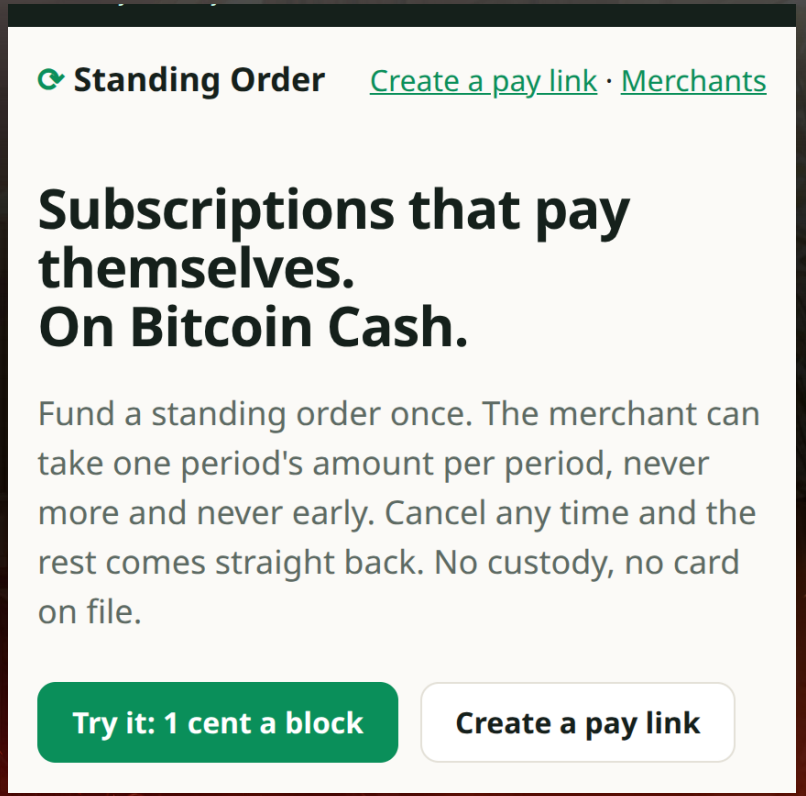
Standing Order

Team: Swift Bot Labs



standing-order.swiftbotlab.workers.dev

swiftbotlab

swiftbotlabs

Standing Order

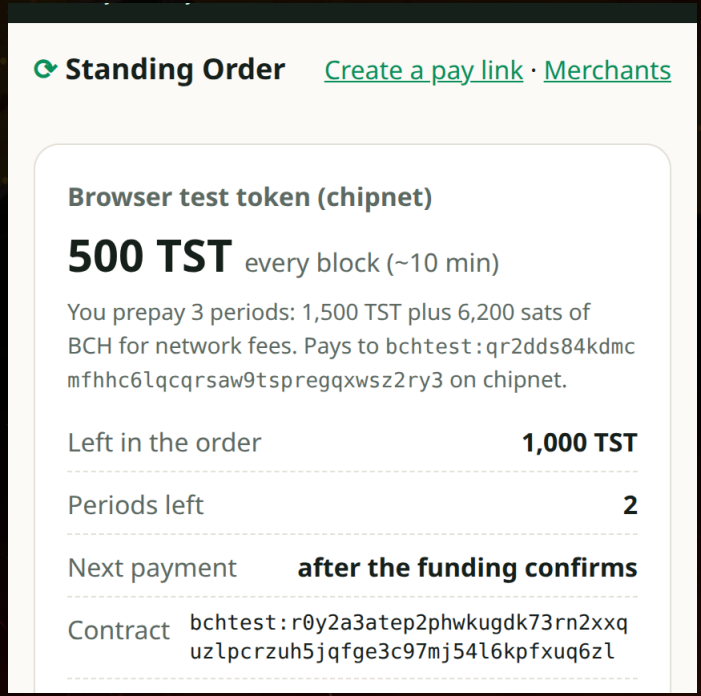
Subscriptions that pay themselves, on Bitcoin Cash

Source: github.com/swiftbotlabs/standing-order

The Story of Standing Order

Recurring payments are how most of the internet gets paid: software, newsletters, memberships, donations. On Bitcoin Cash every payment is a push, so the subscriber has to remember, every period, forever.

We wanted card-on-file convenience without the custody: the money stays in a contract the subscriber controls, and the merchant can only take what was agreed, when it was agreed.



Benefits

- Merchants get paid on schedule, collected automatically
- Subscribers prepay once and cancel any time
- No custody: no one can move the funds outside the rules
- Price in BCH or in PUSD, so a $5 plan stays $5
- Anyone can trigger a due payment, so it keeps working without us
- WizardConnect wallets: scan, approve, done

BCH Usage

Each order is a covenant written in SolidCash (Solidity syntax compiled to BCH Script). claim() pays the merchant exactly one period's amount after a one-period relative timelock; the change must return to the same covenant, so it can never pay early or twice.
cancel() needs the subscriber's signature. CashTokens: the same rules for PUSD.

```
function claim() {
    // At most one claim per period: the UTXO must be at least `period` blocks old.
    require(this.age >= period);
    // Output 0 always pays the merchant.
    require(tx.outputs[0].lockingBytecode == new LockingBytecodeP2PKH(merchant));
    int currentValue = tx.inputs[this.activeInputIndex].value;
    int changeValue = currentValue - amountPerPeriod - maxFee;
    if (changeValue < amountPerPeriod + maxFee) {
        // Last claim: what is left (minus at most maxFee) goes to the merchant; the order ends.
        require(tx.outputs[0].value >= currentValue - maxFee);
    } else {
        // Regular claim: exactly one period's amount, and the change stays in this contract.
        require(tx.outputs[0].value == amountPerPeriod);
        require(tx.outputs[1].lockingBytecode == tx.inputs[this.activeInputIndex].lockingBytecode);
        require(tx.outputs[1].value >= changeValue);
    }
}
```

The Vision

- Pay links today; a checkout button and merchant dashboard next
- Annual plans, usage-based plans, team seats
- Stablecoin payroll and allowances on the same covenant
- Target: a Subscribe button on every BCH creator page




THA YOU!