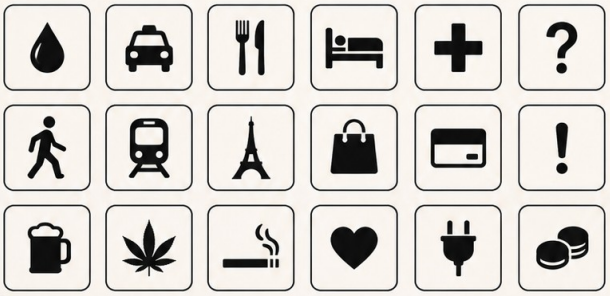
Pictiq

A VISUAL PROTOCOL FOR SHORT MESSAGES

Icons-as-tiles for universal,
offline-friendly communication



CORE LEXICON | CONTEXT PACKS | PORTABLE LAYOUTS | REAL-WORLD USE

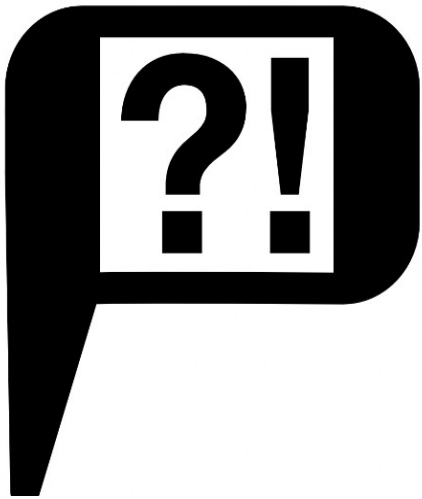
A SMALL PROTOCOL FOR BIG IDEAS

by Anton Biletskyi-Volokh

v1.0.0-CORE

Pictiq

A Visual Protocol for Short Messages



Icons-as-tiles for universal, offline-friendly communication

Pictiq is an open visual protocol and icon lexicon for short messages across language barriers.

Source repository

GitHub: **markoblogo/pictiq**

Release: **v1.0.0-core**

Trademarks and brands

Pictiq avoids logos, wordmarks, and trademarked brand identity elements in its icon sets. Any brand-related icon inclusion requires explicit rights and maintainer approval.

Table of Contents

Preface

Most communication systems assume you have time, context, and a shared language.

Real life often does not.

Travel is the obvious case: you're tired, offline, in a hurry, and you need to find something simple — water, a restroom, a taxi, a hospital. But the same "language gap" shows up in many other places: crowded events, noisy streets, emergencies, quick signage, stickers, and even everyday situations where speaking isn't practical.

We already trust one global visual language: **signage**. Road signs, airport pictograms, and safety symbols work because they are minimal, standardized, and readable at a glance. They do not try to say everything — they try to say the few things that matter, clearly.

Pictiq is an attempt to extend that idea into a **small, usable protocol** for short messages. It is not a full language. It does not aim to translate novels. Instead, it aims to be:

- **Fast:** point to a tile, get the meaning across.
- **Offline-friendly:** no connectivity required.

- **Composable:** a few tiles form a short phrase.
- **Printable:** it must survive real-world printing on fabric, paper, plastic, and stickers.
- **Systematic:** rules are explicit, so the set can grow without breaking style or meaning.

This handbook is a compact introduction: how to read Pictiq, what the core lexicon contains, how context packs work (including a Paris example), and what kinds of products and experiments become possible once the system is consistent.

If you only remember one idea, remember this:

a tile is a message. Pointing is already a sentence.

Chapter 1 — Why Pictiq?

1.1 A small problem that happens everywhere

You do not need a dramatic story to understand why short messages matter.

Most of the time, the problem is painfully ordinary.

You are tired. The street is loud. Your phone battery is low. The network is unstable.

You are in a place where your words do not land — not because people are unfriendly, but because the shared context is missing.

You need something simple:

- water
- a restroom
- a taxi
- a place to sleep

- a way to pay
- help

The question is not *“how do I translate a paragraph.”*

The question is *“how do I communicate a single intention in five seconds.”*

Pictiq starts from this kind of moment.

It treats communication as a constrained environment: short time, weak context, high cost of confusion.

That constraint is not a weakness. It is a design brief.

1.2 We already trust one global visual language

There is a reason road signs work.

They do not try to describe the world.

They represent a small set of actions and objects that matter in motion.

They are:

- minimal
- standardized
- readable at a glance
- robust across materials (metal, plastic, paint, print)
- robust across cultures (not perfect, but surprisingly effective)

Signage is the proof that a “limited language” can still be useful.

It is not a poetic system. It is a practical one.

Pictiq borrows that mindset: do fewer things, but do them reliably.

1.3 From icon shirts to a protocol

Long before Pictiq became a repository, it existed as a simple idea: ***what if you could wear a small dictionary.***

A shirt, a bag, a sticker — anything that can carry a set of symbols — becomes a portable communication board.

The interaction is simple: you point. The other person understands or at least guesses correctly.

But a design like this has a problem: without rules, it becomes a pile of pictograms.

The moment you try to expand it — add cities, add scenarios, **add new “words” — inconsistency creeps in:**

- different drawing styles
- unclear boundaries between similar meanings
- icon sets that cannot be combined cleanly
- symbols that look fine on screen but fail in print

- brand and trademark risks
- no stable identifiers for tooling

At that point the project is no longer “a shirt design.”

It is a language engineering problem, just in a very small domain.

Pictiq is my decision to approach the idea systematically:

as a **protocol** with a fixed visual grammar, stable IDs, and reproducible assets.

1.4 What Pictiq is (and what it is not)

Pictiq is a **minimal visual protocol for short messages**.

It is designed for:

- pointing-based communication
- quick signage and stickers
- offline-friendly scenarios
- small phrases that fit into 1–5 tiles
- machine-readable metadata (IDs, packs, simple rules)

Pictiq is **not**:

- a full natural language
- a writing system for long texts
- a universal replacement for existing pictograms
- a substitute for translators in rich conversation

This is not a limitation to apologize for.

It is the reason the system can stay consistent.

A protocol works when it is strict.

1.5 The design philosophy: zero-intent

Most constructed “icon languages” start by building grammar.

Pictiq starts earlier: at the gesture.

If you point at an object tile, the default meaning is already present:

- *“this”*
- *“I need this”*
- *“where is this”*

Pictiq treats that as the baseline mode of reading.

Then it adds a tiny layer of operators that amplify meaning without turning the protocol into a full grammar system.

In Pictiq terms:

- *punct_question* turns a tile into a question
- *punct_exclaim* turns it into urgency / insistence
- *logic_no* negates the previous tile

Example pattern:

- *need_water*
- *need_water* + *punct_question*
- *need_water* + *punct_exclaim*
- *need_water* + *logic_no*

The point is not precision. The point is speed and clarity under constraints.

1.6 Why it matters beyond travel

Travel is the simplest sales pitch, but it is not the only one.

Short visual protocols have value wherever language becomes expensive:

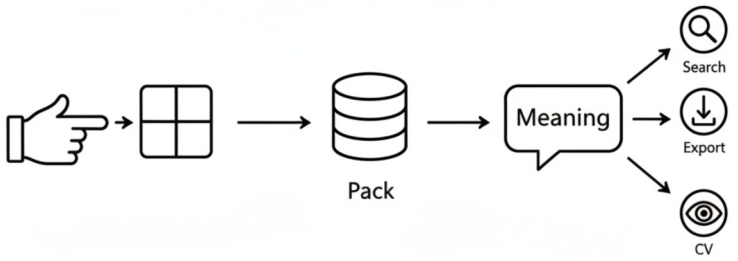
- loud environments (clubs, festivals, stadiums)
- noisy machinery / industrial settings
- accessibility scenarios (communication support, quick prompts)
- emergency situations
- physical objects that “speak” without screens (labels, tags, keycards)
- signs that must be understood instantly

Pictiq also has a second life: it is friendly to automation.

Because every tile has a stable identifier and the system has explicit reading modes (grid vs line), **it becomes:**

- indexable
- searchable
- convertible (SVG PNG)
- scannable (computer vision)
- representable as structured data for tools and LLM workflows

This handbook is not a technical paper, but the direction is clear: ***a stable visual protocol is a useful interface layer between humans, objects, and machines.***



1.7 What this handbook will give you

This book is a field guide, not a manifesto.

You will get:

- the rules of the protocol
- the core lexicon
- example phrases you can use immediately
- a concrete city pack example (Paris)
- portable communication objects — shirts, wallet cards, lighters, luggage tags, and personal screens
- a short comic and a small poetic experiment
- a brief look at where this can go next (CV/ML/LLMs)

If you only want the practical minimum, you can read the protocol rules and the core grid, and stop there.

If you want to build with Pictiq — packs, stickers, merch, and experiments — the later chapters show how the system scales.

Chapter 2 — The Protocol

2.1 The smallest workable rule set

Pictiq is intentionally strict. The protocol is not a loose icon collection.

It is a minimal system designed to stay readable across:

- print materials (paper, fabric, plastic, stickers)
- sizes (from a phone screen to a shirt)
- contexts (travel, signage, micro-merch, quick notes)
- tools (search, export, and future recognition)

This chapter defines the rules that make the system coherent.

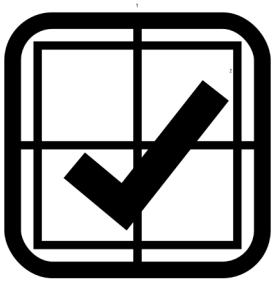
If you learn only one thing: **a tile is a message**.

Everything else is a controlled way to amplify that message.

2.2 Rule 1 — Tiles (the basic unit)

Every lexicon entry is a **tile**: a pictogram inside a rounded-square frame.

- The frame is mandatory for all lexicon tiles.
- The frame creates a “word boundary” in any layout.
- The **Pictiq logo is not a tile** and must not use the tile frame.



Practical consequence:

When people see a sequence of tiles, they read it as a Pictiq phrase.

When they see a single tile on an object, they treat it as a standalone message.

2.3 Rule 2 — Zero-intent (default reading)

Pictiq does not require an explicit “I want / I need / where” operator by default.

Pointing at an object tile is already meaningful.

Examples:

- *need_toilet*

  → “toilet / I need a toilet / where is the toilet”

- *place_hotel*

  → “hotel / lodging / where to stay”

- *comm_wifi*

  → “Wi-Fi / internet / where is Wi-Fi”

This is the core design decision.

It keeps phrases short and makes the system usable without prior training.

2.4 Rule 3 — Punctuation tiles amplify meaning

Two punctuation tiles provide the strongest “grammar” with minimal complexity:

- *punct_question* — turns a tile into a question (“where / how / is there / can I”)
- *punct_exclaim* — urgency / insistence (“urgent / help / attention”)

Examples:

- *need_water* + *punct_question*

  → “Where can I get water?”
- *place_hotel* + *punct_question*

  → “Where is a hotel?”
- *safety_medical* + *punct_exclaim*

  → “Medical help! Urgent!”
- *need_toilet* + *punct_exclaim*

  → “Urgent — toilet!”

Note: punctuation tiles do not replace context. They shape it.

A question tile asks, an exclaim tile escalates.

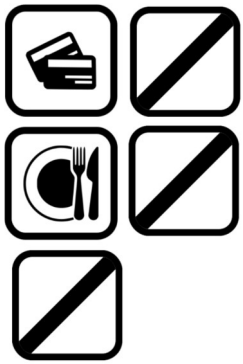
2.5 Rule 4 — Negation uses a dedicated tile

Negation is expressed as:

- X + *logic_no*

  And *logic_no* can also be a standalone answer.

Examples:



- *money_card* + *logic_no*

  → “No card / card not accepted”

- *need_food* + *logic_no*

  → “No food / not hungry / don’t want food” (context decides)

- *logic_no*

  → “No.”

This is intentionally blunt. In real use, bluntness helps.

Fixed negations (context packs)

Context packs may include *fixed negations* as single tiles (object with the same diagonal slash), for stable personal constraints such as “no alcohol” or “no meat.”

Core does not include them.

2.6 Rule 5 — Quantity tiles (minimal set)

Pictiq uses a minimal quantity system:

- *qty_1*
- *qty_2*
- *qty_5*
- *qty_plus* (more / another)
- *qty_minus* (less / reduce)

Placement rule: quantity follows the object.

Examples:

- *need_water + qty_2*

  → “Two waters.”

- *need_food + qty_1*

  → “One portion.”

- *need_water + qty_5 + qty_5*

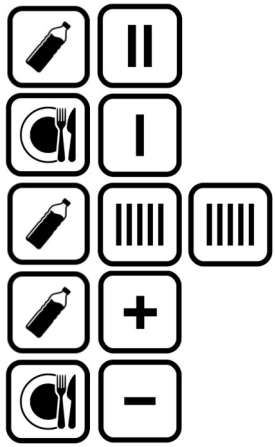
  → “Many waters.” (allowed repetition)

- *need_water + qty_plus*

  → “More water.”

- *need_food + qty_minus*

  → “Less food / smaller portion.”



In live face-to-face communication, people often use fingers for exact numbers.

Quantity tiles are most useful for signage, distance, or compact phrases.

2.7 Rule 6 — Compounds (BASE + QUALIFIER)

Two adjacent object tiles can form a compound:

- Left tile: **base concept**
- Right tile: **qualifier**

Examples:

- *place_shop* + *comm_phone*

  → “phone / electronics shop”

- *place_shop* + *need_food*

  → “grocery / food shop”

- *move_car* + *service_tools*

  → “car service / repair”

Compound guidance:

Keep compounds short and obvious. In core usage, one compound is usually enough.

2.8 Rule 7 — Phrase length

Pictiq is optimized for short phrases.

- Optimal: **1–3 tiles**
- Maximum (protocol limit): **5 tiles**

Examples (5 tiles max):

- *move_public* + *time* + *punct_question*

  → “Public transport schedule / when?”

- *move_taxi* + *money_card* + *punct_question*

  → “Can I pay for a taxi by card?”

When you need multiple sentences, write multiple lines (one phrase per line).

2.9 Representations: Grid mode vs Line mode (spread)

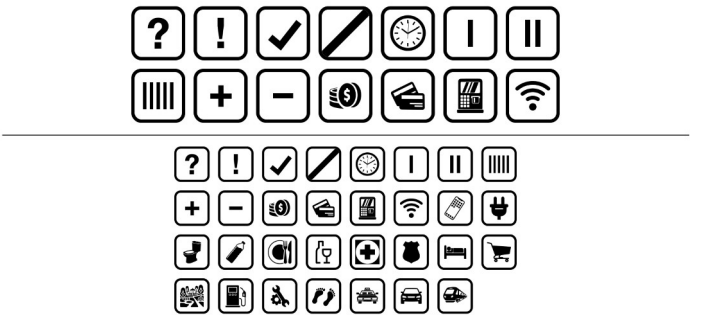
Pictiq supports two reading modes. They use the same lexicon, but different presentation.

Grid mode (catalog / pointing board)

A grid is designed for **pointing**.

It works best on shirts, bags, printed cards, posters, and as a quick “vocabulary board.”

- Each tile is a selectable “word.”
- A grid can be split into sections (context pack on top, core at the bottom).
- A person points to one or more tiles to form a message.



Line mode (phrase line)

A line is designed for **sending and reading** as a short phrase.

- Tiles are placed left to right.
- The sequence is read as one message.
- Best for stickers, labels, chat exports, captions, and simple signage.

Examples:

- *need_toilet + punct_question*
- *safety_police + punct_exclaim*
- *need_water + qty_2 + money_coins*

2.10 Why these constraints exist

Pictiq's constraints are not aesthetic preferences.

They are reliability constraints.

Why tiles?

Because tiles provide **explicit boundaries**.

- They keep phrases readable in messy real-world layouts.
- They separate Pictiq from decorative icon art.
- They allow search, indexing, and stable IDs in tools.

Why no text inside SVG icons?

Because text breaks reproducibility.

- Different fonts render differently.
- Text introduces localization inside the icon (which defeats the purpose).
- Text is fragile at small print sizes.

Even punctuation and tally marks are stored as curves, not <text>.

Why currentColor only?

Because the same SVG must work everywhere:

- black-on-light
- white-on-dark
- any background color (fabric, plastic, paper)

Using currentColor keeps the asset monochrome and invertible without editing geometry.

Why max 5 tiles?

Because beyond that, readability collapses.

At 1–3 tiles, Pictiq stays fast and practical.

At 5 tiles, it is still usable.

Beyond that, you are building a language the protocol was not designed to be.

Pictiq chooses strictness over completeness.

2.11 Quick reference (rules in one box)

If you want a single summary to remember:

- **Tile = message** (zero-intent)
- Ask with: + punct_question
- Urgency with: + punct_exclaim
- Negate with: + logic_no
- Quantity follows the object
- Compounds are: BASE + QUALIFIER
- 1–3 tiles is best, 5 max
- Grid = pointing board, Line = phrase

Chapter 3 — Core Lexicon

3.1 The Core Lexicon (and how to practice it)

The Core Lexicon is the stable baseline of Pictiq.

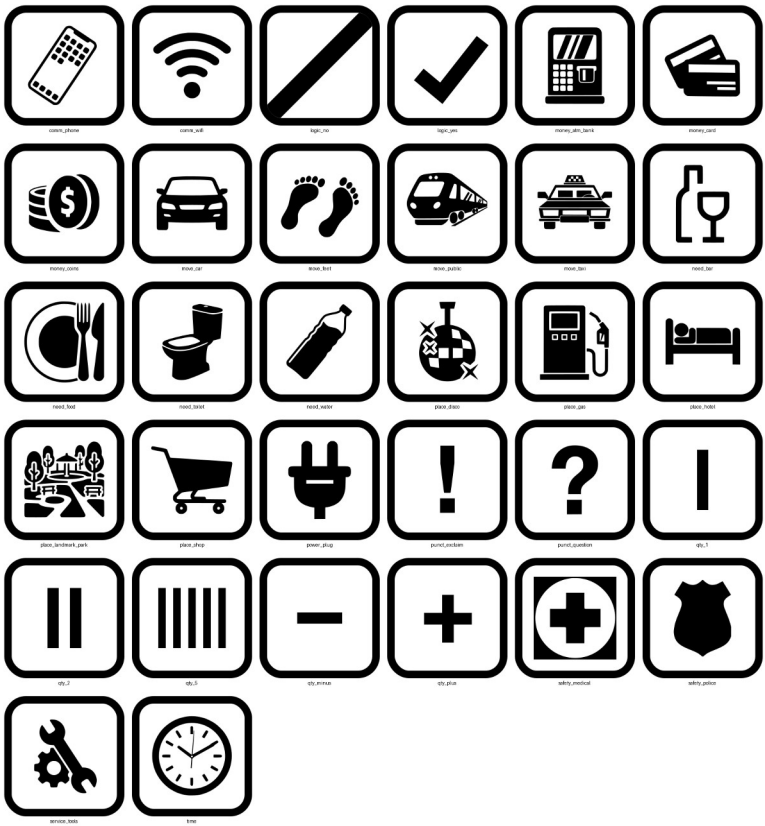
It is deliberately small. The goal is not completeness — the goal is reliability.

Core contains the concepts that show up everywhere:

- basic needs (water, food, toilet)
- movement and transport
- money and payment
- safety and assistance
- a small set of operators (question, urgency, negation, quantity)

Core is designed to work in two modes:

- **Grid mode:** a pointing board you can wear or print
- **Line mode:** short phrases you can write, export, or place on signage



A five-minute practice method

If you want to “learn” Pictiq quickly, do not memorize the whole grid.

Practice **patterns** instead:

1. pick any object tile
2. add a question: + punct_question
3. add urgency: + punct_exclaim
4. negate it: + logic_no
5. add quantity: + qty_2 or + qty_plus

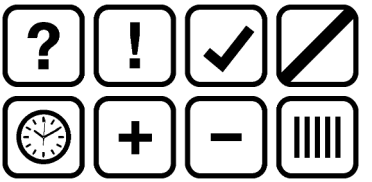
In other words: learn what the operators do, and Core becomes usable fast.

3.2 Punctuation & Logic (the operator layer)

Pictiq’s grammar is intentionally thin.

Instead of many function words, it relies on a small operator layer that amplifies meaning.

This layer makes the protocol feel “alive” without becoming a full language.



What each operator does (in practice)

Question

- *punct_question* turns a tile into “where / how / can I / is there”.

Examples:

- *need_toilet* + *punct_question*
- *place_hotel* + *punct_question*
- *comm_wifi* + *punct_question*

Urgency / insistence

- *punct_exclaim* escalates the same tile into urgency.

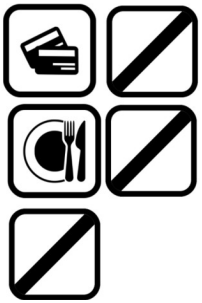
Examples:

- *safety_medical* + *punct_exclaim*
- *need_toilet* + *punct_exclaim*

Negation

- *logic_no* negates the previous tile (and can also be a standalone answer).

Examples:

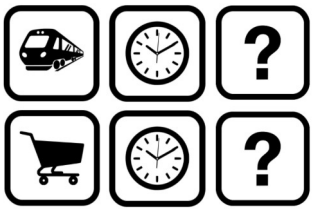


- *money_card* + *logic_no*
- *need_food* + *logic_no*
- *logic_no*

Time

- *time* is used for schedule and availability questions.

Examples:



- *move_public* + *time* + *punct_question*
- *place_shop* + *time* + *punct_question*

Ready-to-use phrase patterns

These patterns cover a large percentage of real-world use:

- *X* + *punct_question* → “Where/how is X?”
- *X* + *punct_exclaim* → “Urgent: X!”
- *X* + *logic_no* → “No X / not X”
- *X* + *time* + *punct_question* → “What time / schedule for X?”

3.3 Quantity (small set, big payoff)

Quantity in Pictiq is minimal by design.

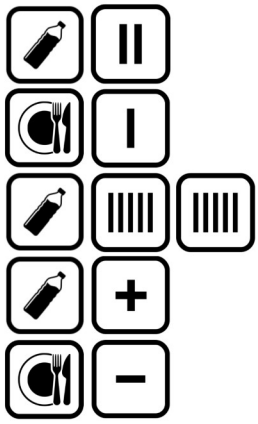
The system avoids “exact arithmetic” and instead supports practical amounts.

Rules

- Quantity follows the object: **WHAT → HOW MUCH**
- *qty_5* may repeat to mean “many”
- *qty_plus* means “more / another”
- *qty_minus* means “less / reduce”

Examples (ready-to-use)

- *need_water + qty_2*
- *need_food + qty_1*
- *need_water + qty_5 + qty_5 (many)*
- *need_water + qty_plus*
- *need_food + qty_minus*



Notes on real-world use

In face-to-face interaction, people often use fingers for exact numbers.

Quantity tiles matter most when:

- you communicate at a distance
- you print a phrase on an object
- you want a compact phrase line for signage/stickers

Quantity is not there to replace natural counting.

It is there to keep the protocol usable when counting is not possible.

3.4 Needs (the “I need it now” layer)

Core needs are the tiles people reach for first.

They work well in zero-intent mode, and they become extremely clear once you add question or urgency.

Mini-grid IDs (4 tiles):

- *need_water* 
- *need_food* 
- *need_toilet* 
- *need_bar* 

Notes

- ***need_toilet*** is the most time-sensitive tile in real use.
- ***need_water*** and ***need_food*** are universal, but context may vary (shop vs restaurant vs availability).
- ***need_bar*** is a “night-life anchor”: it is not about cuisine, it is about drinks.

Ready-to-use phrase patterns

Questions:

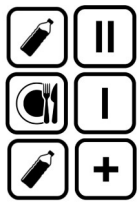
- *need_toilet + punct_question*
- *need_water + punct_question*
- *need_food + punct_question*
- *need_bar + punct_question*

Urgency:

- *need_toilet + punct_exclaim*
- *need_water + punct_exclaim*

Quantities:

- *need_water + qty_2*
- *need_food + qty_1*
- *need_water + qty_plus*



Negation (context-dependent):

- *need_food* + *logic_no*

 “No food / not hungry / don’t want food” (context clarifies)

3.5 Money & Payment (when meaning must be explicit)

Money tiles are special: they often carry operational meaning (pay, exchange, card accepted).

In practice, they reduce confusion more than almost any other category.

Mini-grid IDs (3 tiles):

- *money_coins*
- *money_card*
- *money_atm_bank*

Notes

- *money_coins* is the most general “money” concept in the protocol.
- *money_card* is extremely practical: many real situations depend on whether card is accepted.

- *money_atm_bank* is for cash access and bank/ATM discovery.

Ready-to-use phrase patterns

Can I pay by card?

- *money_card* + *punct_question*

Card not accepted:

- *money_card* + *logic_no*

Where is an ATM / bank?

- *money_atm_bank* + *punct_question*

Cash / money (context decides):

- *money_coins* + *punct_question*

  (Often reads as “money/exchange/where to pay?”)

Buying a thing (implicit):

- *need_food* + *money_coins*
- *move_taxi* + *money_card* + *punct_question*

3.6 Movement (get there, move now)

Movement tiles act as both nouns and actions.

They are among the best examples of Pictiq's "zero-intent" approach: the vehicle often implies the verb.

Mini-grid IDs (4 tiles):

- *move_feet* 
- *move_public* 
- *move_taxi* 
- *move_car* 

Notes

- *move_feet* can mean "walk", "go there on foot", or simply "go".
- *move_public* is intentionally generic: it covers bus/metro/train as "public transport".
- *move_taxi* is "private transport now".
- *move_car* can mean "car / drive / rental" depending on context.

Ready-to-use phrase patterns

Where is public transport / how to use it?

- *move_public* + *punct_question*

Schedule / when does it run?

- *move_public* + *time* + *punct_question*

Taxi + card?

- *move_taxi* + *money_card* + *punct_question*

Walk there?

- *move_feet* + *punct_question*

Car service / repair:

- *move_car* + *service_tools*

3.7 Places & Services (where things are)

Places are the “anchors” of Pictiq.

They answer the most common real-world question: *where is X?* — even when you never explicitly say “where”.

Mini-grid IDs (5 tiles):

- *place_hotel*
- *place_shop*
- *place_landmark_park*
- *place_gas*
- *service_tools*

Notes

- *place_hotel* is not “sleep” as an internal state — it is **lodging** as a destination.
- *place_shop* is intentionally generic and becomes specific through compounds.
- *place_landmark_park* is a deliberately broad “go see / walk around” anchor.
- *place_gas* and *service_tools* support the reality of movement: fuel and repair are universal needs once vehicles are involved.

Ready-to-use phrase patterns

Where is a hotel?

- *place_hotel* + *punct_question*

Hotel with Wi-Fi?

- *place_hotel* + *comm_wifi* + *punct_question*

Where is a shop?

- *place_shop* + *punct_question*

Grocery / food shop:

- *place_shop* + *need_food*

Phone / electronics shop:

- *place_shop* + *comm_phone*

Where is a landmark / park?

- *place_landmark_park* + *punct_question*

When is it open? (practical proxy using time)

- *place_shop* + *time* + *punct_question*

Fuel / gas station:

- *place_gas* + *punct_question*

- *move_car* + *place_gas*

Repair / service:

- *service_tools* + *punct_question*

- *move_car* + *service_tools*

3.8 Safety, Communication & Power (stay connected, stay safe)

This section is where the protocol becomes most “real.”

These tiles are not decorative — they are the ones you want available when things go wrong.

Mini-grid IDs (5 tiles):

- *safety_medical*
- *safety_police*
- *comm_wifi*
- *comm_phone*
- *power_plug*

Notes

- safety_medical is intentionally broad: doctor, hospital, medical help.
- safety_police covers police/security/authority in a generic, non-national way.
- comm_wifi and comm_phone are the modern survival layer: connection and a phone.

- power_plug matters more than people expect: no power means no translation, no maps, no calls.

Ready-to-use phrase patterns

Medical help, urgent:

- *safety_medical* + *punct_exclaim*

Where is medical help?

- *safety_medical* + *punct_question*

Police / security, urgent:

- *safety_police* + *punct_exclaim*

Where is police?

- *safety_police* + *punct_question*

Wi-Fi?

- *comm_wifi* + *punct_question*

Wi-Fi in a hotel?

- *place_hotel* + *comm_wifi* + *punct_question*

Phone / call?

- *comm_phone* + *punct_question*

Phone shop (compound):

- *place_shop* + *comm_phone*

Power / charging?

- *power_plug* + *punct_question*

No power / not available:

- *power_plug* + *logic_no*

  (useful on signage; in conversation the context will usually clarify)

3.9 Putting Core together (quick patterns)

Core becomes easy when you think in patterns, not in words.

The three most universal constructions

1. **Find a thing**

- X + punct_question

2. **Make it urgent**

- X + punct_exclaim

3. **Negate**

- X + logic_no

A few “core-only” phrase lines that cover a lot of reality

- *need_toilet + punct_question*
- *need_toilet + punct_exclaim*
- *need_water + qty_2*
- *money_card + punct_question*
- *money_card + logic_no*

- *move_public + time + punct_question*
- *place_hotel + comm_wifi + punct_question*
- *safety_medical + punct_exclaim*

3.10 Core vs Context (the boundary that keeps the system clean)

Core is meant to stay stable.

It is the set you can put on a shirt today and still recognize years later.

Context packs are where the protocol expands safely:

- cities (Paris, Tokyo, London)

- scenarios (food allergies, hiking, festivals)
- personal packs (your own constraints and interests)
- corporate packs (venues, events, internal signage)

Context packs may add:

- less universal objects
- region-specific landmarks
- “fixed negations” as single tiles
- specialized vocabulary that would bloat Core

This boundary matters because it protects the protocol from turning into an icon dump.

End of Chapter 3

You now have everything you need to use Pictiq Core as a communication board: ***operators, needs, places, movement, money, and safety.***

Next, we move from the baseline into a real-world demonstration:

Chapter 4 — Context Packs

starting with a concrete example: **Paris** (a city pack used in a wearable layout: context on top, core on the bottom).

Chapter 4 — Context Packs (Paris)

4.1 Why packs exist

Core is designed to be stable and universal.

But real situations are never universal.

A city, a sport, a job, a medical restriction, a festival — each context introduces meanings that do not belong in Core. If we keep adding everything into Core, the protocol turns into a noisy icon dump: harder to print, harder to learn, harder to maintain.

Context packs solve this by separating two responsibilities:

- **Core:** a small, stable baseline you can rely on anywhere.
- **Packs:** focused additions for a specific context, designed to be mixed with Core.

A pack is not a second language.

It is a thin layer of extra tiles that makes Core feel locally fluent.

4.2 The pack rule: add only what Core cannot say

A strong pack follows one strict rule:

A new tile is added only if the meaning cannot be expressed by Core tiles or a simple compound.

In practice, packs add three types of content:

1. **Anchors:** highly recognizable objects or places you can point at.
2. **Experience markers:** what people actually do there (a “theme park” is a good example).
3. **Reusable additions:** sometimes a pack reveals a missing generic concept. When this happens, the icon becomes reusable and later benefits many packs.

This rule keeps packs small and prevents style drift.

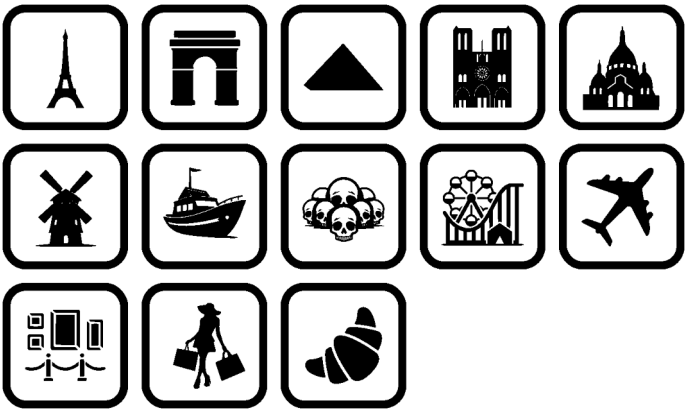
4.3 Paris Pack v0.1 as a working example

Paris is a perfect first city pack: globally recognizable, heavily visited, and it demonstrates the whole idea of contextual navigation.

Paris Pack v0.1 contains a compact set of tiles that people actually point at:

- a few iconic landmarks
- a few “city experience” anchors
- a few reusable travel concepts that also work outside Paris

No text. No logos. No brand wordmarks. Only silhouettes.



Paris-specific vs reusable icons

The pack includes both:

- **Paris-specific tiles** (unique silhouettes that strongly index the city)
- **Reusable tiles** (generic concepts useful across many cities)

Reusable tiles are still included in the Paris pack because the pack is also a practical travel board. The difference is in how they are named and reused across packs.

4.4 The canonical merch layout: “top context / bottom core”

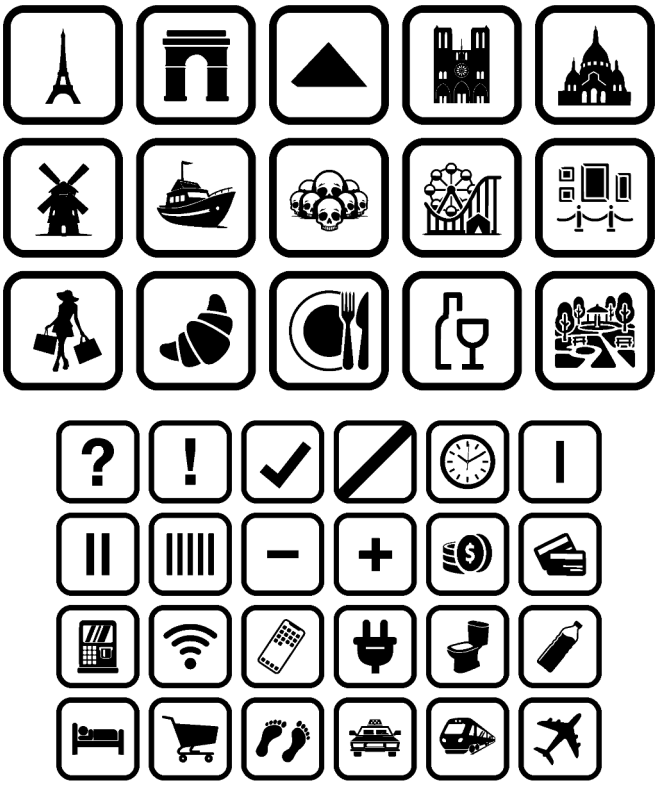
Pictiq started from a physical interaction idea: point at a tile, communicate immediately.

That idea becomes strongest when the layout is optimized for pointing. The canonical wearable layout is:

- **Top block:** context pack tiles (larger)
- **Bottom block:** a practical Core subset (smaller)

This layout improves usability:

- the top block is where a traveler points for *where to go / what to see*
- the bottom block covers universal needs and operators: questions, urgency, payment, connection, safety



4.5 The Paris Core subset

A city board does not need the entire Core printed.

Instead, we use a subset optimized for typical city travel:

- punctuation and logic (*punct_*, logic_*, time*)
- money and payment (*money_**)
- connectivity and power (*comm_*, power_plug*)
- essential needs (*need_**)
- basic places (*place_hotel, place_shop, place_landmark_park*)
- movement (*move_**)
- airport as a travel anchor (*place_airport*)

The key point: **Core does not change.**

We are only choosing what to print for this specific product layout.

4.6 Using packs in phrases (line mode)

In real use, most communication is pointing in a grid.

But for documentation, signage, and digital use, Pictiq also supports short phrase lines.

Below are examples written as icon_id sequences.

Landmark navigation

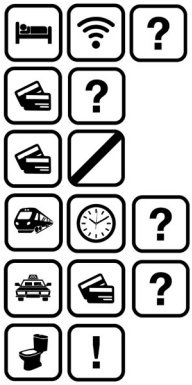
- *paris_eiffel_tower* + *punct_question*
- *paris_arc_de_triomphe* + *punct_question*
- *paris_louvre_pyramid* + *punct_question*
- *paris_notre_dame* + *punct_question*

City experiences

- *place_theme_park* + *punct_question*
- *place_catacombs* + *punct_question*
- *place_art_gallery* + *punct_question*
- *place_fashion_shopping* + *punct_question*

Practical travel

- *place_hotel* + *comm_wifi* + *punct_question*
- *money_card* + *punct_question*
- *money_card* + *logic_no*
- *move_public* + *time* + *punct_question*
- *move_taxi* + *money_card* + *punct_question*
- *need_toilet* + *punct_exclaim*



4.7 Why the Paris pack is intentionally incomplete

A city contains infinite details.

A travel pack should not.

Paris Pack v0.1 is deliberately constrained:

- it contains only the most “pointable” anchors
- it stays readable at arm’s length
- it is printable and robust across products
- it avoids text and brand identity elements

- it remains easy to expand later without breaking the system

If users want more, the correct response is not to bloat the pack.

The correct response is to release **v0.2** or make a separate themed pack (food, museums, nightlife, etc.), each small and focused.

4.8 From pack to products

Once a pack exists as a clean digital asset (SVG + metadata + grid), it becomes reusable product infrastructure.

The same pack can become:

- a t-shirt board (top pack + bottom core subset)
- a tote bag board
- sticker sheets and labels
- a small printed guide page
- machine-readable assets for search, export, and future CV recognition

The pack is not only “for the book.”

It is an asset that scales into many forms.

End of Chapter 4

Core gives you a stable baseline.

Context packs turn it into a system that adapts to real life.

Paris Pack shows what Pictiq looks like when the protocol becomes local: the same Core stays stable, while a small set of contextual tiles makes the system immediately useful in a specific place. This is the practical reason packs exist: not to expand the language endlessly, but to make communication boards sharper, faster to point at, and easier to print without noise. Once a pack is defined as SVG + metadata, it becomes a reusable asset: it can live in a book, on a shirt, in stickers, or inside a search tool.

In the next chapter, we move from context packs to physical and digital surfaces. The question is no longer only which symbols belong together, but how the same profile can move between a shirt, a wallet card, a lighter, a luggage tag, or a screen.

Chapter 5 — Carrying the Language

5.1 From a vocabulary to an object

A visual protocol becomes more useful when it leaves the screen. A Pictiq tile can live on a shirt, a card, a lighter, a luggage tag, or a phone. The underlying vocabulary does not change. What changes is the selection of tiles and the physical surface on which they appear.

This led to a simple separation:

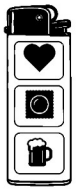
Vocabulary → Profile → Layout → Object

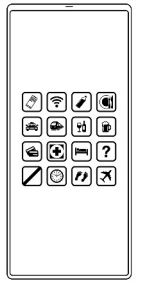
The vocabulary contains the available concepts.

A **profile** selects the concepts that matter in a particular situation. A **layout** determines how those concepts fit a particular surface. The object provides the final layer of context.

This separation is important because it prevents every product from becoming a separate design project. The same Paris profile can be rendered as a shirt or a wallet card. The same wallet-card layout can carry Paris, nightlife, or another future profile.

The language stays stable. The object changes.





5.2 One profile, different surfaces

The Paris examples provide the simplest demonstration.

The shirt developed in the previous chapter uses two visual levels:

- larger primary tiles for the most relevant Paris concepts;
- smaller secondary tiles for supporting communication.

A wallet card uses exactly the same principle, but the physical constraint is different.

Instead of placing both groups on one surface, the card separates them between two sides:

Side A: 12 primary tiles, arranged 3 × 4.

Side B: 20 secondary tiles, arranged 4 × 5.

The card itself follows standard payment-card proportions: 85.60 × 53.98 mm. Nothing fundamental about the Paris vocabulary changes. Only its physical representation does.

This is an important property of the protocol.

A context pack should not have to be redesigned every time it moves to a new medium. Once the concepts have been selected and prioritized, layouts can transform the same profile into different physical objects.

5.3 One layout, different contexts

The reverse is also true.

A layout can remain stable while its content changes.

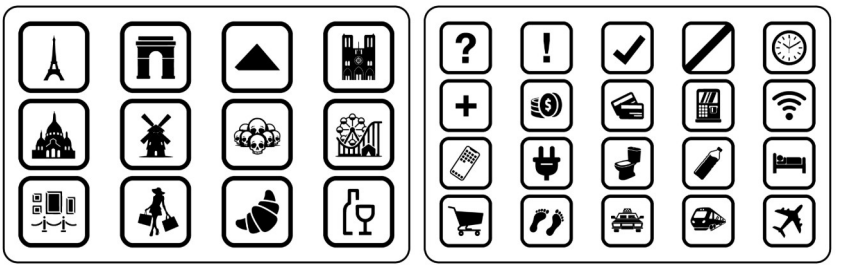
The Paris Wallet Card and the Nightlife Wallet Card have identical physical rules:

- the same dimensions;
- the same 3 × 4 primary side;
- the same 4 × 5 secondary side;
- the same visual hierarchy.

But their profiles are completely different.

Paris prioritizes landmarks, city experiences, transport, food, and practical travel.

Nightlife prioritizes bars, clubs, accommodation, taxi, smoking, drinks, romance, safer sex, and a small set of operators.



This makes a layout similar to a container.

The container does not define the message. The profile does.

A future Tokyo card, hiking card, festival card, food-allergy card, or personal accessibility card can use exactly the same physical specification without requiring a new visual system.

5.4 When the object already provides context

Sometimes the physical object itself tells us what kind of vocabulary is useful.

A lighter is an obvious example.

There is little reason to place thirty generic travel symbols on it. The fact that the person is holding a lighter already narrows the probable context.

For the Nightlife profile, six tiles are enough.

One side carries:

`item_cigarette`

`item_cannabis`

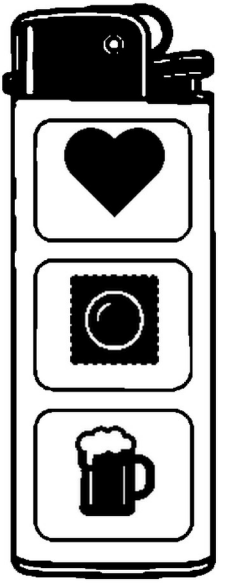
`need_bar`

The other carries:

`love_heart`

`item_condom`

`drink_beer`



The cigarette tile illustrates an important Pictiq principle particularly well.

By itself, it represents the cigarette/smoking concept.

Pointing to it in different situations may mean:

“Can I smoke?”

“Where is the smoking area?”

“Where can I buy cigarettes?”

If smoking is prohibited, there is no need for a second dedicated “no smoking” word:

```
item_cigarette + logic_no
```

The object, location, gesture, and surrounding tiles complete the interpretation.

Pictiq deliberately uses context rather than trying to encode every possible sentence as a separate pictogram.

5.5 Purpose-built communication

Other objects have their own natural vocabulary.

Consider a luggage tag.

A traveler moving through airports and stations repeatedly needs a small cluster of concepts:

- airport;
- public transport;
- taxi;
- walking;
- hotel;
- payment;
- time;

- question.

A luggage tag can therefore become more than identification attached to a suitcase. It can also become a small transport communication board.

Again, the complete Core is unnecessary.

The purpose of the object determines which part of the protocol deserves physical space.

This suggests a useful design rule:

Do not ask how many icons can fit. Ask which icons earn the space.

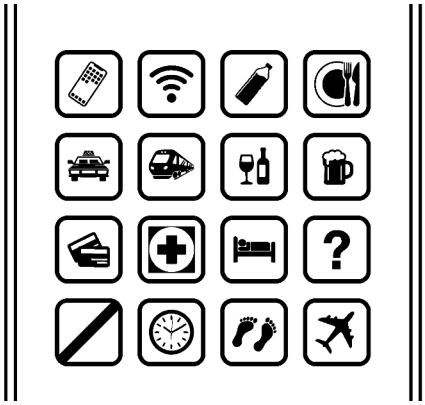
More vocabulary is not automatically better communication.

5.6 The personal layer

Physical objects are only one possibility.

A smartphone lock screen is already a surface that people carry everywhere.

A Pictiq lock screen can contain a personal selection of concepts without requiring an application, network connection, login, or even an unlocked phone.



The demonstration profile mixes communication, needs, movement, nightlife, money, safety, and accommodation.

But there is no reason for two people to use the same selection.

One person might prioritize:

- medical assistance;
- hotel;
- taxi;
- water;
- Wi-Fi.

Another might prioritize:

- food;
- public transport;
- bar;
- payment;
- airport.

A third could build a profile around a specific accessibility requirement, hobby, profession, or trip.

At this point Pictiq stops behaving like a fixed phrasebook and starts behaving like a configurable interface.

The lexicon remains shared.

The selection becomes personal.

5.7 Primary and secondary

The experiments with shirts and wallet cards produced another practical rule.

Not all concepts deserve equal visual weight.

Pictiq profiles can therefore distinguish between two groups:

Primary

The concepts most relevant to the context. They receive more space and are easier to point at.

Secondary

Supporting concepts that remain useful but do not need the same prominence.

On a shirt, primary tiles can appear larger at the top while secondary tiles appear smaller below.

On a wallet card, primary tiles occupy the first side and secondary tiles the second.

Other layouts may represent the distinction differently.

There is one strict rule:

The same tile should not appear in both groups.

Repeating a tile wastes scarce physical space without adding meaning.

5.8 Artwork, preview, and mockup

Building physical layouts also revealed that three different kinds of images should not be confused.

Artwork

The exact flat graphic intended for printing or display.

A phone wallpaper is artwork.

The two sides of a wallet card are artwork.

The three tiles printed on one side of a lighter are artwork.

Preview

A schematic representation showing where that artwork belongs on the physical object.

A phone outline with the wallpaper inside it is a preview.

A line drawing of a lighter carrying three tiles is a preview.

The preview should use the exact generated artwork rather than redraw or reinterpret its icons.

Mockup

A realistic presentation image.

A person wearing a Paris Pictiq shirt near the Eiffel Tower is a mockup.

A photographed lighter lying on a bar counter could be another.

Mockups are useful for presentation and commerce, but they are not part of the protocol itself.

This distinction makes the production process reproducible:

Profile → Artwork → Preview → optional Mockup

5.9 Build less, reuse more

The purpose of profiles and layouts is not to create an endless catalog of products.

It is the opposite.

A small number of reusable components should produce many useful combinations.

A city profile can become:

- a shirt;
- a wallet card;
- a phone screen.

A nightlife profile can become:

- a wallet card;
- a lighter;

- a sticker.

A travel profile can become:

- a luggage tag;
- a card;
- a printable sheet.

And a personal profile can combine existing vocabulary without adding a single new icon.

The system grows through recombination before it grows through vocabulary.

That is an important constraint.

Before creating a new icon, ask whether an existing tile or compound can express the idea.

Before creating a new profile, ask whether an existing profile can be adapted.

Before creating a new layout, ask whether an existing physical format already solves the problem.

5.10 Carry your own language

Pictiq began with a simple interaction:

point at a symbol.

Portable layouts extend that interaction without changing it.

The communication surface can be a shirt, a piece of plastic in a wallet, a luggage tag, a lighter, or a screen. Each object contributes context. Each profile contributes relevance. The shared lexicon keeps the system recognizable.

This is why Pictiq does not need to become a large language to become more useful.

It can remain small while appearing in more places.

And once the same visual protocol can move between people, objects, and screens, another possibility appears: machines can read it too.

The next chapter explores the less practical and more experimental edge of Pictiq: visual storytelling, poetry, computer vision, and communication with language models.

Chapter 6 — Experiments & Machines

6.1 Beyond utility

So far, Pictiq has stayed deliberately practical.

The protocol was designed around situations where communication needs to be short, fast, and good enough: finding water, asking about payment, locating transport, pointing at a landmark, or carrying a small personal vocabulary on an object.

That constraint is important. Pictiq is not intended to replace natural language.

But once a visual protocol has a stable vocabulary, explicit rules, canonical graphics, and machine-readable IDs, an obvious question appears:

How far can we push it before it stops working?

This chapter contains a few experiments rather than new protocol rules.

Can Pictiq carry a familiar narrative?

Can it translate poetry?

Can a camera recognize it?

Can the same visual message become structured input for a language model or another machine?

The failures are as interesting as the successes. A small protocol becomes easier to understand when we deliberately test its boundaries.

6.2 A comic without words

Comics already combine several communication systems.

A character's expression carries information. Composition tells us where to look. Sequence creates time. Objects establish context. Speech bubbles add only the part that the image cannot communicate efficiently.

This makes comics an unusually natural environment for Pictiq.

To test this, we adapted a famous exchange from Lewis Carroll's *Alice's Adventures in Wonderland*: Alice encounters the Cheshire Cat while trying to decide which way to go.

The original conversation is philosophical as much as practical. Alice asks which road she should take. The Cat replies that the answer depends on where she wants to go. Alice does not particularly care. In that case, the Cat suggests, the direction does not matter very much.

A literal Pictiq translation would be a bad idea. The protocol has no words for *depends*, *somewhere*, *provided that*, or *which way ought I to go*.

But the comic does not need them.

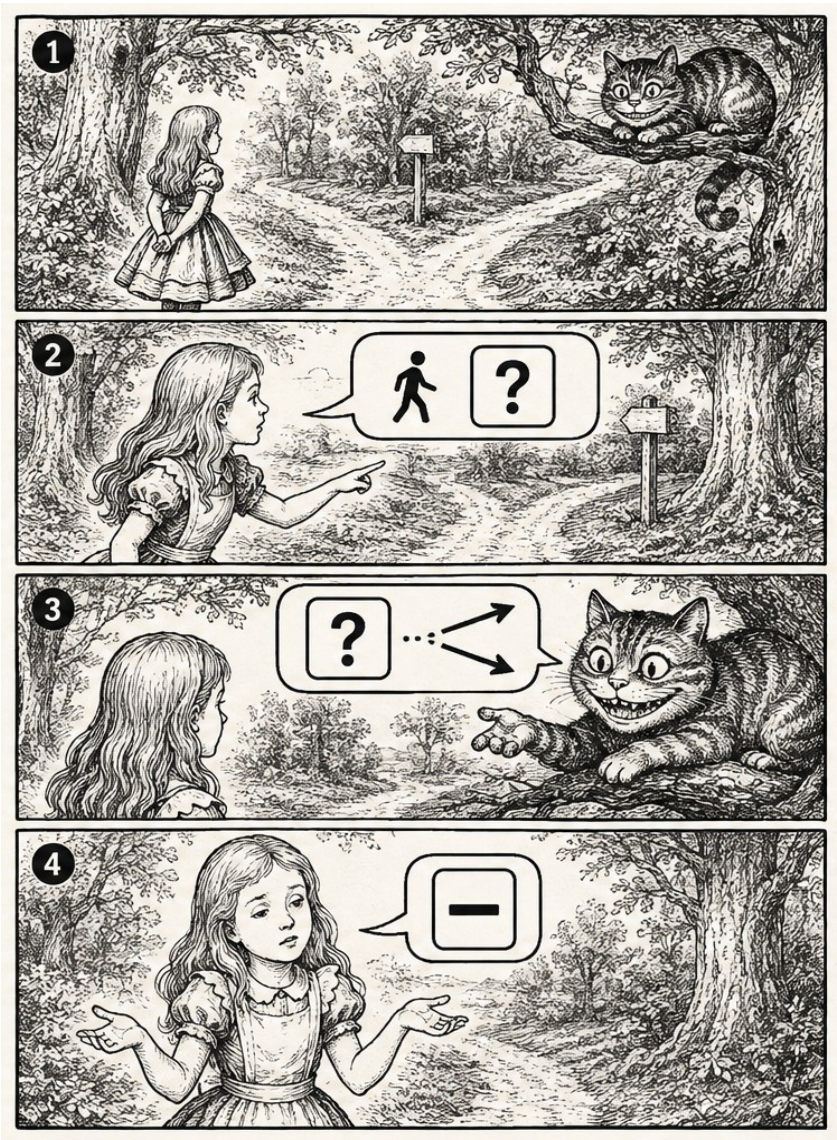
The fork in the road supplies **choice**.

Alice's gesture supplies **uncertainty**.

The Cheshire Cat supplies the other participant.

The sequence of panels supplies **before and after**.

Pictiq only has to carry the missing semantic fragments.


1
2
?
3
?
4
—

For example:

`move_feet + punct_question`

can function as:

Which way should I go?

The Cat can return the question largely through visual context:

`punct_question`

Alice can answer:

`logic_no`

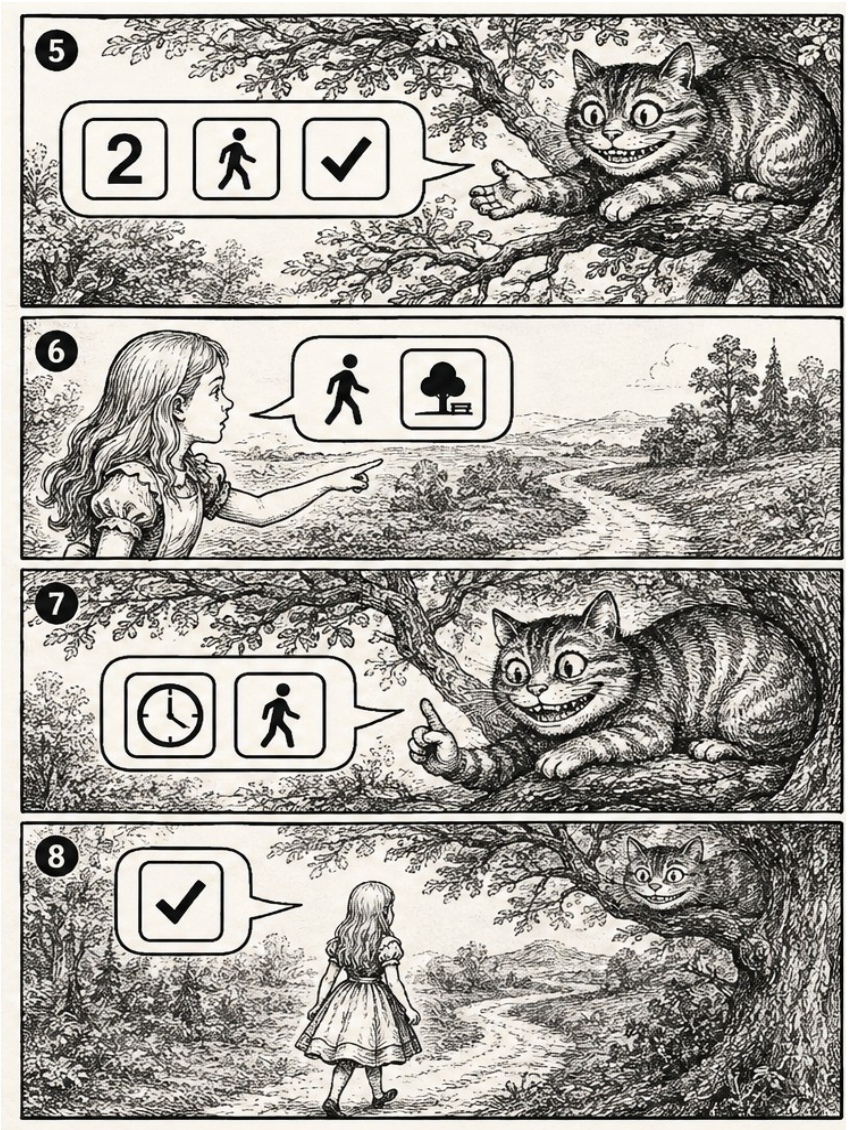
And two possible paths can be reduced to:

`qty_2 + move_feet + logic_yes`

Later:

`time + move_feet`

is enough to preserve the basic idea that continued movement eventually leads somewhere.


5
2
6
7
8

None of these sequences reproduces Carroll's sentences.

That is precisely the point.

The experiment demonstrates a distinction between **translation** and **communication**.

Pictiq is weak at reproducing syntax, voice, irony, and philosophical nuance. But when images and circumstances already carry those things, a surprisingly small symbolic vocabulary can complete the message.

In other words, the comic works not because Pictiq has enough words.

It works because it knows when it does not need them.

6.3 Poetry and impossible translation

Poetry is almost the opposite problem.

In ordinary practical communication, losing some nuance may not matter. If someone understands that you need a taxi or cannot pay by card, the protocol has succeeded.

In poetry, nuance is often the content.

Rhythm matters. Sound matters. Metaphor matters. Ambiguity matters. A word can carry historical, emotional, and cultural associations that no pictogram can reproduce.

So poetry is a deliberately unfair test for Pictiq.

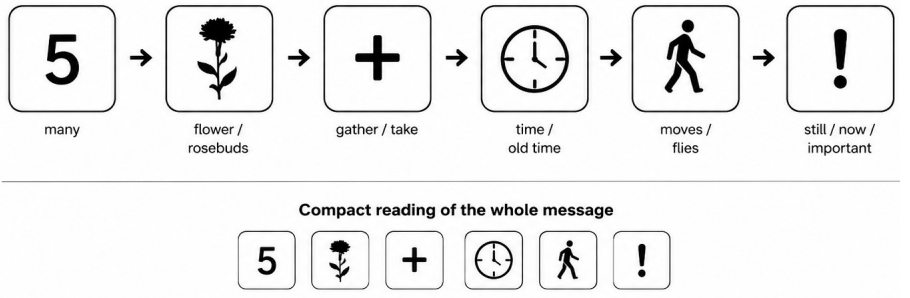
For the experiment, we use the opening of Robert Herrick's *To the Virgins, to Make Much of Time*:

> ***Gather ye rosebuds while ye may,***
> ***Old Time is still a-flying;***

The lines compress several ideas into very little space: flowers, gathering, opportunity, passing time, youth, urgency, and the personification of Time itself.

Pictiq cannot preserve all of that.

But it can attempt to preserve the structural core.



The experimental reading uses:

`qty_5 + nature_flower + qty_plus`

followed by:

`time + move_feet + punct_exclaim`

The first sequence suggests:

many flowers → gather/take more.

The second suggests:

time → moves → urgent.

This is not Herrick translated into another language.

It is closer to **semantic compression**.

What survives is the broad instruction:

Take the flowers while you can. Time is moving.

What disappears is substantial: the direct address, rhythm, rhyme, the image of rosebuds, the personification of Old Time, the cultural associations of *carpe diem*, and much of the emotional force of the original.

That loss tells us something useful about Pictiq.

The protocol performs best when the objective is to preserve **actionable meaning**.

It performs poorly when the objective is to preserve **literary form**.

That is not a defect to fix by adding thousands of icons. It is a boundary worth keeping visible.

6.4 When machines can see the tiles

There is another reason for making Pictiq systematic rather than treating it as a collection of illustrations.

Every canonical tile has an identity.

A person sees a pictogram.

Software can see:

`need_water`

or:

`move_taxi`

or:

`logic_no`

Once the visual object has been mapped to a stable ID, image recognition and semantic interpretation become separate problems.

A hypothetical computer-vision pipeline could be extremely simple:

camera → tile recognition → icon_id → profile/context → meaning or action

A vision system does not necessarily need to infer from scratch that a particular drawing represents water.

It needs to recognize that the image corresponds to the canonical Pictiq tile `need_water`.

The lexicon already supplies the semantic layer.

This opens several experimental possibilities.

A camera could scan a printed Pictiq phrase and reconstruct its IDs.

A phone could recognize tiles on a card or shirt and display corresponding meanings in another language.

A physical sign could become both human-readable and machine-readable without adding a QR code to every message.

A robot or kiosk could recognize a small controlled vocabulary much more reliably than arbitrary drawings.

The restricted vocabulary becomes an advantage.

General visual understanding is difficult because the world contains almost unlimited variation. Pictiq deliberately creates a tiny visual universe with canonical forms, explicit identifiers, and known combinations.

The machine does not have to understand everything.

It only has to understand Pictiq.

No such recognition system is required for the protocol to work. A printed card remains useful without electricity, software, or a network connection.

Machine recognition is an additional layer, not a dependency.

6.5 Pictiq as structured input for language models

The same separation between representation and meaning becomes even more interesting with language models.

Consider this Pictiq phrase:

```
need_water + qty_2 + punct_question
```

A human can see three tiles.

A program can represent exactly the same message as:

```
[need_water, qty_2, punct_question]
```

At that point the image itself is no longer necessary.

The sequence is structured data.

A language model could receive those IDs together with the protocol rules and context:

```
profile: travel

sequence: [need_water, qty_2,
punct_question]
```

From that representation it could produce an appropriate natural-language interpretation:

Can I get two waters?

or, depending on context:

Where can I get two bottles of water?

The reverse operation is also possible.

A user might type:

Can I pay for the taxi by card?

A system could attempt to reduce the sentence to:

```
move_taxi + money_card + punct_question
```

This is more interesting than simply asking an AI to generate pictograms.

The icons are only one rendering of an underlying protocol.

The same message can exist as:

- visual tiles for a person;
- IDs in JSON;
- a short phrase for a language model;
- recognized objects for computer vision;
- commands or states for another software system.

That makes Pictiq potentially useful as a small intermediary representation between humans and machines.

6.6 Prompts, agents, and physical systems

Once messages are represented as stable IDs, other experiments become possible.

A Pictiq sequence could become a constrained prompt.

Instead of giving an agent an unrestricted natural-language request, an interface might offer a controlled visual vocabulary. The resulting sequence is smaller, easier to validate, and less ambiguous at the protocol level.

A physical device could expose only the concepts it supports.

A hotel kiosk might load a hospitality profile.

A festival terminal might load a festival profile.

A robot operating in a public environment might expose movement, assistance, destination, and safety tiles.

Personal profiles could provide a small communication interface tailored to an individual rather than requiring everyone to navigate the entire vocabulary.

This does not mean pictograms are inherently better than words.

Natural language is vastly more expressive.

The interesting possibility is different:

sometimes a deliberately restricted interface is useful precisely because it cannot say everything.

Software interfaces already work this way.

Buttons, menus, schemas, API parameters, and command sets reduce possible expression in exchange for predictability.

Pictiq explores what happens when that principle is applied to a tiny human-readable visual protocol.

6.7 Where this could go

Pictiq v1 is intentionally small.

The current project already contains enough infrastructure to test several directions without changing the basic protocol:

Computer vision

Recognition of canonical tiles from cameras, photographs, printed cards, clothing, and signs.

Natural-language translation

Conversion between short natural-language intentions and Pictiq sequences.

Generated context profiles

Creating a compact vocabulary for a city, event, profession, hobby, or personal situation from the existing lexicon before deciding whether new icons are actually required.

Personal communication interfaces

Wallet cards, lock screens, wearables, accessibility aids, and other surfaces built around a person's own priorities.

Robotics and physical interfaces

Using the same tiles as human-facing controls and machine-readable identifiers.

Prompting and agents

Treating Pictiq IDs as a small constrained command vocabulary.

New physical layouts

Applying profiles to objects whose physical context already reduces ambiguity.

Some of these ideas may work.

Some probably will not.

That is part of the project.

Pictiq does not need every experiment to succeed. The purpose of a protocol is to create a stable foundation on which experiments can fail cheaply without forcing the foundation itself to change.

Conclusion — A Small Protocol

Pictiq began with a very simple observation:

sometimes you do not need a shared language.

You need one shared symbol.

A person can point to water.

A traveler can point to a taxi.

A card can ask about payment.

A shirt can become a communication board.

A lighter can carry six concepts selected by the context of the object itself.

A phone can carry a personal vocabulary.

And a camera may eventually recognize the same tiles that a person sees.

The project deliberately stops before becoming a complete language.

That boundary is important.

Natural languages are powerful because they can express almost anything. Pictiq is useful for almost the opposite reason: it tries to express a small class of messages with as little friction as possible.

Its basic architecture is therefore simple:

Lexicon → Profile → Layout → Object

And underneath every visual tile is something even simpler:

a stable ID.

That allows the same protocol to exist simultaneously as graphics, printed objects, structured data, and potentially machine-readable input.

The current vocabulary is not final.

The current packs are not final.

The layouts are experiments.

What matters is that the rules connecting them are explicit enough for the system to grow without becoming an arbitrary collection of icons.

The smallest useful Pictiq message remains the one we started with:

a tile is a message.

Pointing can be enough.

Everything after that is composition.

Resources

Pictiq is an open visual-protocol project under active development.

The repository contains the current protocol specification, canonical SVG icons, Core lexicon, context profiles, layouts, validation tools, printable assets, and releases.

Project repository
GitHub: `github.com/markoblogo/pictiq`

Core release
`v1.0.0-core`

Available materials

- protocol specification
- Core lexicon
- canonical SVG icons
- Paris context pack
- printable grids
- wallet-card layouts
- portable-layout examples
- source files and validation tools

License

Commercial use requires a separate agreement. See `COMMERCIAL_LICENSE.md` in the repository.

What comes next

Pictiq is intended to continue as an experimental ecosystem rather than end with this handbook.

Planned experiments include additional context packs, printable communication objects, books built around Pictiq, and a collection of “speaking” merchandise that turns ordinary objects into small communication surfaces.

Further experiments may explore computer vision, automatic profile generation, translation, language models, and physical interfaces.

For new releases, packs, books, merchandise, and experiments, follow the project repository.

Download it. Print it. Wear it. Point at it. Build something strange with it.