E X E R C I C E T E C H N I Q U E

Cadastre & Parcelles

Extraction et visualisation automatique de parcelles cadastrales

TypeScript • PDF • cadastre.gouv.fr

Contexte

Tu travailles dans une agence immobilière / un cabinet de géomètres. Les collaborateurs doivent régulièrement extraire des plans cadastraux depuis cadastre.gouv.fr et y mettre en évidence les parcelles concernées par un dossier.

Aujourd'hui, ce processus est entièrement manuel : téléchargement du PDF, ouverture dans un éditeur, tracé à la main… L'objectif est d'automatiser tout cela en TypeScript.

Objectifs pédagogiques

- Manipuler des fichiers PDF en TypeScript (lecture, écriture, dessin)
- Interagir avec une API publique (cadastre.gouv.fr)
- Gérer des données géographiques (coordonnées, polygones, projections)
- Structurer un projet progressivement par niveaux de difficulté
- Produire un livrable visuel propre et utile

Stack recommandée

Outil	Usage
TypeScript + tsx / ts-node	Langage principal et exécution
pdf-lib	Manipulation et génération de PDF
axios ou fetch	Appels HTTP vers l'API cadastre
API cadastre.gouv.fr	Récupération des données parcellaires (GeoJSON)
proj4 (optionnel)	Conversion de coordonnées (Lambert 93 → pixels)

Les niveaux

Level 1 — Dessiner une parcelle sur un PDF fourni

🎯 Objectif

L'utilisateur fournit un PDF cadastral + un identifiant de parcelle. Le programme dessine le contour de la parcelle sur le PDF et génère un nouveau fichier.

📥 Entrées

- Un fichier PDF cadastral (téléchargé manuellement depuis cadastre.gouv.fr)
- Un identifiant de parcelle (ex : 33063000BW0012)

📤 Sortie

- Un fichier PDF avec le contour de la parcelle dessiné en surbrillance (contour rouge + remplissage semi-transparent)

✅ Critères de validation

- Le PDF de sortie s'ouvre correctement
- La parcelle est visuellement identifiable
- Le contour correspond à la vraie géométrie de la parcelle

Level 2 — Récupérer automatiquement le plan cadastral

🎯 Objectif

L'utilisateur ne fournit plus le PDF. Il donne uniquement l'identifiant de la parcelle. Le programme va chercher le plan cadastral correspondant sur cadastre.gouv.fr, puis y dessine la parcelle.

📥 Entrées

- Un identifiant de parcelle uniquement

📤 Sortie

- Un fichier PDF téléchargé automatiquement + annoté

✅ Critères de validation

- Le PDF est téléchargé automatiquement sans intervention manuelle
- Le plan contient la bonne zone géographique
- La parcelle est correctement dessinée

Level 3 — Plusieurs parcelles adjacentes

🎯 Objectif

Supporter la sélection de plusieurs parcelles adjacentes. Le programme doit adapter la vue pour englober toutes les parcelles et les dessiner avec des couleurs différentes.

📥 Entrées

- Une liste d'identifiants de parcelles (au moins 2)

📤 Sortie

- Un PDF unique avec toutes les parcelles mises en évidence

✅ Critères de validation

- Toutes les parcelles sont visibles sur le même plan
- Chaque parcelle a une couleur distincte
- Une légende identifie chaque parcelle

Level 4 — Encart avec surfaces bâties et non bâties

🎯 Objectif

Ajouter au PDF un encart récapitulatif affichant pour chaque parcelle : la surface totale, la surface bâtie, la surface non bâtie, et le pourcentage d'occupation.

📥 Entrées

- Une liste d'identifiants de parcelles

📤 Sortie

- Un PDF annoté avec un encart de synthèse des surfaces

✅ Critères de validation

- L'encart affiche les bonnes surfaces (tolérance de 5%)
- Les zones bâties sont visuellement distinctes sur le plan
- L'encart est lisible et bien positionné (ne masque pas le plan)

Récapitulatif des niveaux

Niveau	Fonctionnalité	Entrées	Difficulté clé
1	Dessiner sur un PDF fourni	PDF + parcelle	Coordonnées géo → PDF
2	Télécharger le PDF automatiquement	Parcelle seule	API cadastre + projections
3	Plusieurs parcelles adjacentes	Liste de parcelles	Bounding box + couleurs
4	Encart surfaces bâties/non bâties	Liste de parcelles	Intersection polygones

Livrables attendus

1. Un repo Git avec un README clair (installation, utilisation)
2. Un fichier par niveau : src/level1.ts, src/level2.ts, src/level3.ts, src/level4.ts
3. Des exemples de PDF générés dans un dossier output/
4. Un fichier de config TypeScript (tsconfig.json) propre

Bonus (facultatif)

- Ajouter des tests unitaires (Vitest ou Jest)
- Gérer les erreurs proprement (parcelle introuvable, PDF invalide…)
- Ajouter un mode --interactive avec inquirer pour choisir la commune et la parcelle
- Générer une image PNG en plus du PDF
- Déployer comme CLI avec un package.json bin

Grille d’évaluation

Critère	Points	Commentaire
Level 1 fonctionnel	/5	
Level 2 fonctionnel	/5	
Level 3 fonctionnel	/4	
Level 4 fonctionnel	/4	
Qualité du code (typage, structure, nommage)	/3	
Gestion d’erreurs	/2	
Documentation / README	/2	

Bonus	/3	
TOTAL	**/28**	

Bon courage ! 🚀